THE DETERMINATION OF REMAINING SATELLITE PROPELLANT USING MEASURED MOMENTS OF INERTIA

THESIS

Jason W. Geitgey, Captain, USAF

AFIT/GAE/ENY/06-J04

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GAE/ENY/06-J04

THE DETERMINATION OF REMAINING SATELLITE PROPELLANT USING MEASURED MOMENTS OF INERTIA

THESIS

Presented to the Faculty

Department of Aeronautics and Astronautics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Aeronautical Engineering

Jason W. Geitgey, BS

Captain, USAF

June 2006

AFIT/GAE/ENY/06-J04

THE DETERMINATION OF REMAINING SATELLITE PROPELLANT USING MEASURED MOMENTS OF INERTIA

Jason W. Geitgey, BS

Captain, USAF

Approved:

__//signed//________________________	__2 Jun 06__
Dr. Richard G. Cobb (Chairman)	Date
__//signed//________________________	__2 Jun 06__
Dr. Bradley S. Liebst (Member)	Date
__//signed//________________________	__2 Jun 06__
Paul A. Blue, Major, USAF (Member)	Date

AFIT/GAE/ENY/06-J04

Abstract

This research is designed to demonstrate that a change in satellite propellant can be determined using measured moments of inertia (MOI) from a satellite. Because satellites are currently incapable of being refueled in orbit it is important to have multiple methods to determine the remaining fuel onboard. This research can also support satellite operator selection of control-system gains to improve performance or recover the spacecraft from an undesirable orbit. To meet the research objectives, new mathematical models of the Air Force Institute of Technology's Simulated Satellite (SimSat) were developed. These models were created using dynamic response analysis techniques on the reaction wheel and SimSat systems. The models were than validated against the existing SimSat hardware. Using a least-squares parameter estimation technique, the model and hardware data were compared to determine the resulting change in measured MOI. Then, using a calibrated baseline model, telemetry data was compared to the model to determine the MOI of the unknown system. The research found it is possible to determine the change in satellite fuel from measured MOI. The research also found there are limits to this detection technique based on the accuracy of the mathematical model and the angle of the detection maneuver being performed.

For my Wife and Daughter

Acknowledgments

First and foremost, I have to thank my loving wife for all of her sacrifices, patience, advice, and support throughout my academic studies. I would like to express my true appreciation for my faculty advisor, Dr. Cobb who always steered me in the right direction no matter how many times he answered the same question. I owe my deepest gratitude to my family and friends who have supported me throughout this academic program. Last, but not least, I have to thank my daughter for helping me to see the humor in almost every situation.

Jason W. Geitgey

.

Table of Contents

Page

Page

List of Figures

Page

Page

List of Tables

Page

List of Symbols and Abbreviations

$\mathbf{a}_i$	Acceleration of a particle
AFIT	Air Force Institute of Technology
$\mathbf{b}_1$	Body axis 1
$\mathbf{b}_2$	Body axis 2
$\mathbf{b}_3$	Body axis 3
deg	Degrees
dm	Differential mass element
dt	Time derivative
DOF	Degrees-of-freedom
e	Signal error
$\dot{e}$	Time derivative of signal error
$\mathbf{F}_{\text{ext}}$	External force
$\mathbf{F}_i$	Force acting on a particle
$\mathbf{F}_{\text{int}}$	Internal force
g	Gravity
$\mathbf{H}$	Angular momentum
$\hat{i}$	Unit vector
$\mathbf{I}_{\text{rw}}$	Moment of Inertia of reaction wheel
$\mathbf{I}_{\text{sat}}$	Moment of Inertia of satellite
$\mathbf{I}_{\text{xx}}$	Moment of Inertia about the x axis
$\mathbf{I}_{\text{xz}}$	Product of Inertia
$\mathbf{I}_{\text{xy}}$	Product of Inertia
$\mathbf{I}_{\text{yy}}$	Moment of Inertia about the y axis
$\mathbf{I}_{\text{yx}}$	Product of Inertia
$\mathbf{I}_{\text{yz}}$	Product of Inertia
$\mathbf{I}_{\text{zz}}$	Moment of Inertia about the z axis
$\mathbf{I}_{\text{zx}}$	Product of Inertia
$\mathbf{I}_{\text{zy}}$	Product of Inertia

$\hat{j}$	Unit vector
J	Cost function
$\hat{k}$	Unit vector
K_p	Proportional gain constant
K_d	Derivative gain constant
M	Total mass of body
$\mathbf{M}$	Moment or applied torque
m_i	Mass of a particle
MOI	Moments of Inertia
n	Number of data points
N	Number of particles
NASA	National Aeronautics and Space Administration
p	Design parameter
PD	Proportional Derivative
PID	Proportional plus Integral plus Derivative
$\mathbf{r}_{cm}$	Center of mass for a rigid body
$\mathbf{r}_i$	Position vector
rad	Radians
SimSat	Simulated satellite
T_d	Time derivative constant
$\bar{\theta}$	Measured data
θ	Modeled data
$\mathbf{v}$	Velocity
$\dot{\omega}$	Angular acceleration
$\vec{\omega}$	Angular velocity
ω_f	Flywheel velocity
ω_1	Satellite angular velocity
ω_{rw}	Angular velocity of reaction wheel

ω_{sat}	Angular velocity of satellite
$\dot{\omega}_{rw}$	Angular acceleration of reaction wheel
$\dot{\omega}_{sat}$	Angular acceleration of satellite
$\mathbf{x}$	Body component
$\mathbf{y}$	Body component
$\mathbf{z}$	Body component

THE DETERMINATION OF REMAINING SATELLITE PROPELLANT USING MEASURED MOMENTS OF INERTIA

I. Introduction

1.1 Motivation

The Department of Defense, specifically the Air Force, faces an increasingly difficult task of working within an ever shrinking budget. The Air Force has a particularly daunting task of trying to strike a balance between new aircraft and space acquisition programs. The cost of satellite systems is astronomical; it is estimated that the research and development of a new satellite is well over a billion dollars (Walter, 2006). Launching the satellite system ranges from 50 to 450 million dollars depending on size and final orbit (Goldin, 1998). It is imperative to maximize the amount of time a satellite remains in orbit. Furthermore, a functional satellite in the correct orbit generates billions of dollars in revenue for a company or provides time sensitive information to military personnel in the field. An editorial in Purdue University's engineering newspaper estimates that one pound of hydrazine or fuel onboard a satellite that sends signals to pagers or residential TV dishes can generate approximately two and a half million dollars of revenue (Venere, 2001). The premature retirement of a satellite system can be devastating to a company or the military. At this time, satellites are incapable of being refueled in orbit. Therefore, it is crucial to conserve fuel and know the exact amount of remaining propellant onboard the satellite.

Moreover, satellites that do not rely on a booster engine use a large amount of onboard propellant to reach their final geostationary orbits. This same propellant is used to maintain the station keeping of the satellite once it reaches a desired orbit. Therefore, an orbital insertion burn that takes significantly longer than planned or a leak in the satellite fuel tank can result in an unintended imbalance of the satellite's center of gravity. This altered configuration of the satellite may lead to a delay or total loss of the affected satellite, incurring a substantial revenue loss to the satellite operator and a multimillion dollar insurance claim against the failed satellite (De Selding, 2004). In order to prevent the latter, it is desirable to have several methods for determining remaining fuel within a satellite.

Satellite on-orbit properties and mass properties are synonymous terms to describe a set of parameters that reflect the distribution of mass in a rigid body. "These parameters include moments of inertia, products of inertia, mass, and center of mass" (Peck, 1999:2). Knowledge of the mass properties assists in the planning of efficient thrusting maneuvers and assists in the prediction of nutation effects during and after maneuvers. It can also support operator selection of control-system gains to improve performance or recover the spacecraft from an undesirable orbit. In theory, the mass properties of a satellite can be used to derive the remaining propellant.

However, accurately measuring mass properties on the ground in large and complex spacecraft is generally not practical or feasible. Satellite configurations on the ground are normally not representative of the final on-orbit satellite configurations due to deployable solar arrays, antennas, or other equipment that may be extended or deployed in orbit. These flexible structures used in satellites are designed to operate in a zero gravity (*g*) environment and are not readily testable on earth. Cooper and Wright summarize two main reasons why obtaining large space structure data on the ground is impractical

> First, the structures will be so complex that theoretical finite element type models will not be sufficiently accurate to be relied on in the absence of test data; in particular the modeling of joints between substructures and of damping is extremely difficult. Second, even substructures will usually be so large and flexible that they will require multiple supports if they are tested in a 1*g* environment (Cooper and Wright, 1992:352).

The inability to collect accurate mass property data on the ground to validate developed mathematical models for a satellite limits the effectiveness of the model. In order to obtain more exact mass properties, techniques are needed to measure spacecraft mass properties on-orbit by performing selected maneuvers and then analyzing measured gyroscopic attitude and angular rate information.

Current and previous methods used to determine the remaining propellant quantity for operational satellites include bookkeeping, use of the ideal gas law with pressure/temperature sensors, thermodynamic measurements, and capacitive sensors. However, these formulations are usually configured prior to launch and they include many predictions and assumptions which may not match the actual satellite system in orbit. It is necessary for a satellite operator or analyst to have multiple methods of formulating/deriving mass property data for the satellite system in order to formulate more accurate on orbit models. Then, when a satellite requires orbital correction, the satellite operator's calculations for the required thruster burn time is extremely precise and it will minimize the fuel expended during the burn. This will prolong the useful lifetime of the satellite.

1.2 Research Objectives

The research aim of this thesis is to develop and experimentally validate an alternative method for calculating remaining propellant while a satellite is in orbit. The application of a least-squares parameter estimation technique to model experimental gyroscope data will be used to determine spacecraft mass properties. A description of the simulated satellite (SimSat) experimental hardware is provided in Section 3.2. Several key objectives for the completion of this research are listed below. Additionally, a flow chart (see Figure 1) provides a pictorial representation of the research methodology.

1. Verify the functionality of the recently installed fiber optic gyroscope on SimSat.
2. Create a new mathematical model of SimSat's reaction wheel system.
3. Develop a simplified and complex closed-loop mathematical model of SimSat ensuring the model correlates to the available experimental hardware.
4. Determine if a change in fuel load can be determined using a detection maneuver, moments of inertia (MOI), and the conservation of angular momentum using the SimSat hardware.
5. Characterize the fiber optic gyroscope's rate of drift.

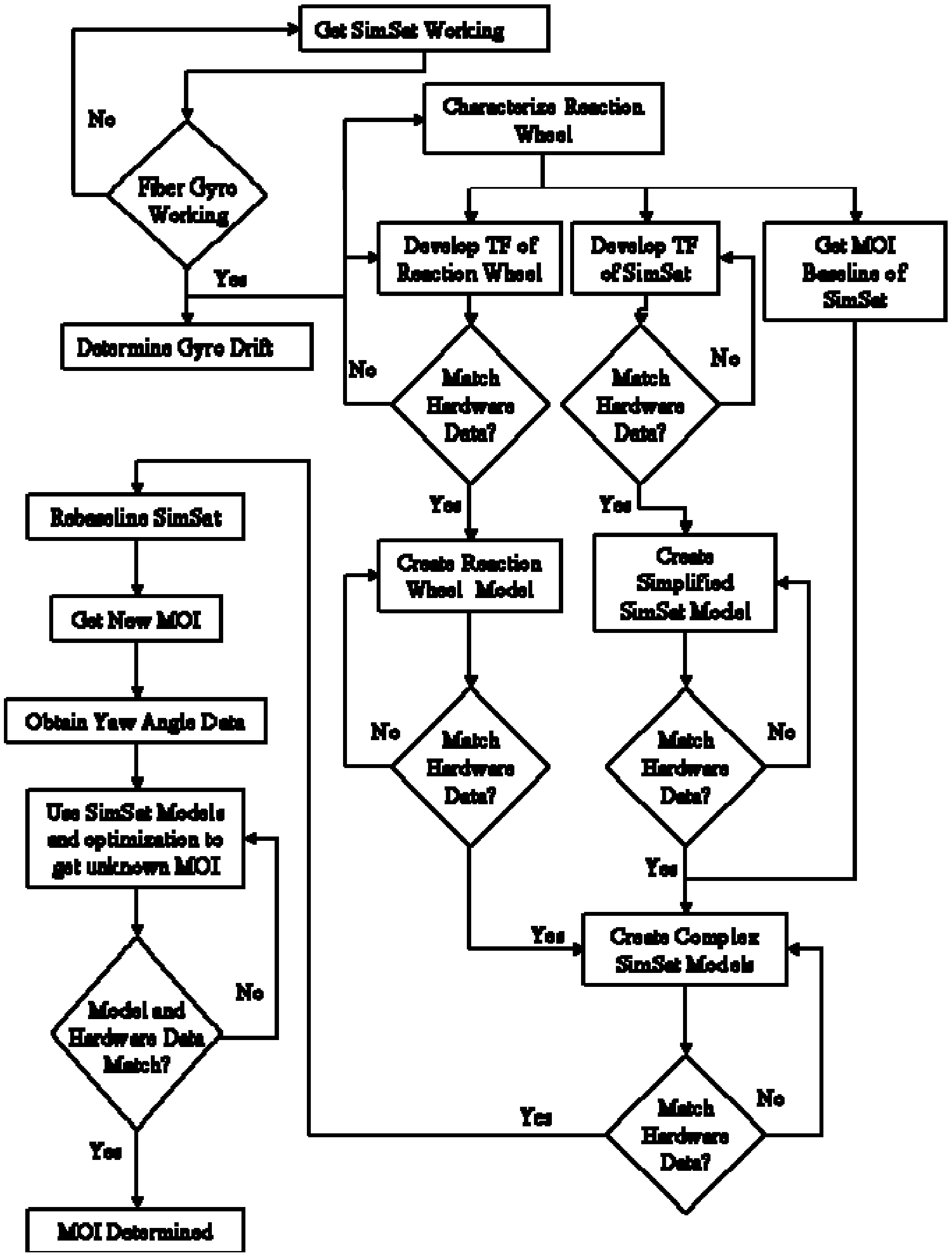


Figure 1. SimSat Research Methodology Flowchart

This research builds upon Smith's addition of the fiber optic gyroscope to the Air Force Institute of Technology's (AFIT) SimSat, see Figure 2 (Smith, 2005: Ch 3,4). It is necessary to ensure this updated system works in order to execute the proposed research. Secondly, SimSat's reaction wheels must be tested and modeled to ensure their performance and any physical limitations are captured in the models. To accomplish objective 3, the math model must be developed and tuned to accurately reflect SimSat's operation. For this research, both SimSat and the resulting model will be treated as a rigid body; SimSat is a fairly stiff structure and the assumption of negligible flexibility is justified. This constraint will ensure the changes to the body can be matched to MOI changes in the system by applying conservation of angular momentum. The latter will permit a direct correlation between MOI and the changing fuel load. Finally, identifying the rate of drift for the fiber optic gyroscope will quantify the amount of error drift can cause in the MOI calculations and resultant fuel estimations.

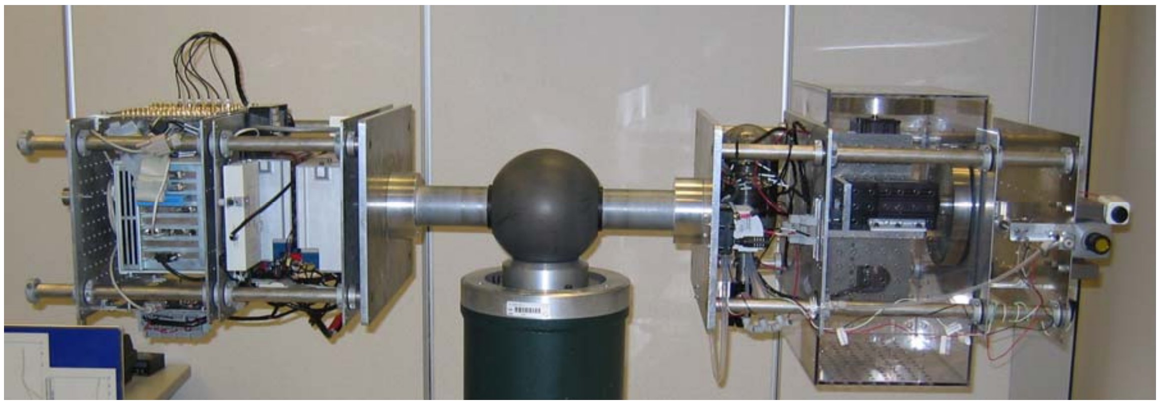


Figure 2. Air Force Institute of Technology's Simulated Satellite (SimSat)

This thesis project has some limitations. The research will not account for the presence of sloshing fuel when determining MOI. Secondly, this thesis is limited to the momentum and the resulting torque generated by reaction wheels rigidly mounted to the body of interest. Finally the use of thrusters, instead of reaction wheels, is not considered and left for future efforts.

1.3 Thesis Outline

The definition of the problem, its relevance to the space community, a brief review of related research, an outline of the research methodology, and the objectives and limitations of this thesis have been discussed in Chapter I. Chapter II provides a review of pertinent research in the field of mass property estimation through the use of MOI and system identification. Additionally, the mathematical theory supporting this research effort is outlined. The approach of the research problem and experimental validation of the analytical methods is described in Chapter III. Chapter IV presents the results and analysis of the various components of this investigation. Finally, Chapter V makes conclusions about the effectiveness of using MOI to determine remaining fuel and suggests future areas of development and study.

II. Background

2.1 Chapter Overview

This chapter reviews relevant research that has been conducted in the field of mass property estimation techniques for satellite systems. This literature review leads into the theoretical foundation that is the basis for this research project. Topics within this chapter include center of mass determination, angular momentum, MOI, Euler's Equations of Motion, rigid body dynamics of SimSat, orienting a satellite system, dynamic response analysis, Proportional Derivative (PD) control, and an optimization technique to determine the change in satellite fuel.

2.2 Relevant Research

The ability to accurately determine satellite mass properties is a fairly extensively studied area of research. The background research heavily utilizes Euler's Equations of Motion and conservation of angular momentum to ascertain spacecraft mass properties through several different methods.

Tanygin and Williams used coasting maneuvers on a three axis stabilized system to estimate the inertia matrix for a spinning spacecraft. The data's applicability is limited because the simulation model does not consider applied torques or firing thrusters (Tanygin and Williams, 1997).

Bergmann, Walker, and Levy used a Gaussian second-order filter to estimate the elements of the inverse inertia matrix and the center of mass location vector. The researchers concluded the mass properties could be estimated to within 1% error when the equations of motion are formulated in the terms of a nonlinear state estimation problem (Bergmann and others, 1987).   Bergmann and Dzielski refine the previous research three years later with a simplified matrix methodology. The researchers applied the algorithm to determine the change in mass properties of a rigid model of the National Aeronautics and Space Administration (NASA) Space Station (Bergmann and Dzielski, 1990).

Conservation of angular momentum instead of the traditional Euler's Equations of Motion is the basis of Peck's approach to determine mass properties of an on orbit, spinning spacecraft. Previous work failed to address unknown energy dissipation. Peck eliminates the uncertainty associated with energy loss when analyzing the mass properties of highly damped systems (Peck, 1999; Peck, 2000). Peck's research findings provide the foundation of this thesis by establishing that conservation of angular momentum is a viable method for determining mass properties for spacecraft.

Using telemetry data from NASA's Cassini spacecraft, Wertz and Lee developed a method to estimate the spacecraft's inertia tensor by conservation of angular momentum. Upon comparison to the Cassini project's book kept method, Wertz and Lee's values were in agreement with ground based tests (Wertz and Lee, 2001).

Dabrowski developed and tested a dynamic detection algorithm to recognize when local parasite masses were added to a satellite. A least squares cost function was used to detect the change in the satellite's moment of inertia, and the resulting addition of the parasitic satellite to the original satellite configuration (Dabrowski, 2003).

This thesis heavily relies on Wertz and Lee, Peck, and Dabrowski's previous work. Wertz, Lee, and Peck showed that conservation of angular momentum can be used to determine mass properties of a satellite. Additionally, Dabrowski demonstrated that moments of inertia can be used to detect a parasitic satellite presence through an increase in MOI. Therefore, these methodologies form the basis to determine the feasibility of detecting the change in fuel on a spacecraft from some initial point utilizing measured data from maneuvers.

2.3 Rigid Body Dynamics

The basis for this research relies on the critical assumption that the experimental hardware constructed is considered a rigid body. This assumption entails the mathematical idealization that the entire structure is rigid, or it will not deform under loads when applied. However, it must be noted that all structures will suffer some type of deformation. This concept of rigidity means if the deformation is negligible compared to the overall body and there are minor elastic properties, then a rigid assumption can be used. Due to the rigid body, the use of classic mechanics and dynamics determine the six degrees-of-freedom (DOF) for the entire rigid body. Translational motion accounts for three DOF and rotational motion accounts for the remaining three. Furthermore, it can be assumed there is no coupling between the translational and rotational DOF. The experimental hardware, SimSat, rests upon an air bearing assembly as shown in Figure 3. The resulting translational equations of motion are absent leaving only the rotational motion of the system to be considered. Before reviewing the rotational equations of motion for a rigid body, also known as Euler's Equations of Motion, it is important to understand how the selection of a body center of mass will impact the rigid body dynamics of the system.

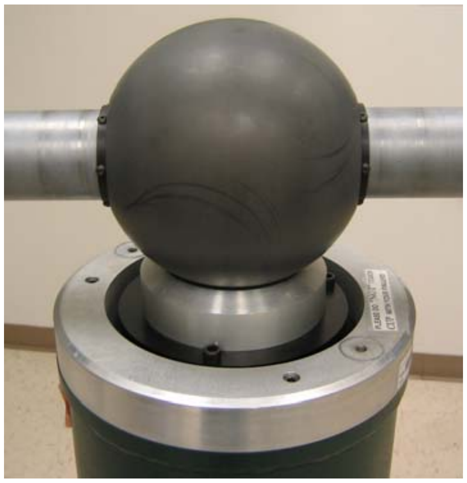


Figure 3. Space Electronics, Inc. Model SE9791 Tri-axis Spherical Air Bearing

2.3.1 *Center of Mass.*

A rigid body is defined as a "body with physical dimensions where the distances between the particles that constitute the body remain unchanged" (Baruh, 1999:323). Thus, a grouping of particles within the rigid body has characteristics defined by each particle's mass and the forces acting upon them. Through the use of Newton's second law and following Halfman's previous work the motion of the group of *N* particles is then

$$\sum \mathbf{F}_i = \sum_{i=1}^{N} \left(m_i \mathbf{a}_i \right) \tag{1}$$

where $\mathbf{F}_i$ is the force acting on the particle, m_i is each particles mass, and $\mathbf{a}_i$ is the acceleration of the particle (Halfman, 1962:130-136). The sum of the forces acting on the particles is then the sum of both the internal and external forces

$$\sum \mathbf{F} = \sum \mathbf{F}_{\text{int}} + \sum \mathbf{F}_{ext} \tag{2}$$

However, due to Newton’s third law where every force has an equal and opposite reactive force, the sum of $\mathbf{F}_{\text{int}} = 0$. This results in only the external forces acting on the particles

$$\sum \mathbf{F}_{ext} = \sum_{i=1}^{N} \left(m_i \mathbf{a}_i \right) \tag{3}$$

By defining a point in three dimensional space where the average “center” of the particle displacements equals zero, the center of mass for a rigid body, $\mathbf{r}_{cm}$, is

$$\mathbf{r}_{cm} = \frac{1}{M} \sum_{i=1}^{N} m_i \mathbf{r}_i \tag{4}$$

where M is the total mass of the body, and $\mathbf{r}_i$ is a vector describing a particle’s displacement from the origin.

Since a rigid body contains an infinite number of particles and the distance is fixed between the particle masses, each particle can be considered a differential mass element denoted as dm. This results in m_i from Equation 4 being replaced with dm and the summations being converted to integrals yielding the center of mass equation for an entire body

$$\mathbf{r}_{cm} = \frac{1}{M} \int_{body} \mathbf{r}_i \, dm \tag{5}$$

Differentiating Equation 5 twice with respect to time and combining it with Equation 3 yields the equation

$$\mathbf{F}_{ext} = M\frac{d^2\mathbf{r}_{cm}}{dt^2} \tag{6}$$

where "the center of mass behaves as if all the mass of the rigid body were concentrated at that point and the total external forces acted there" (Wiesel, 1997:98). Therefore, through the proper application of the center-of-mass concept, the only remaining forces to be considered are the net external forces acting on a rigid body. Forces acting on a rotating body are determined by rotational equations of motion or Euler's Equations of Motion which will be discussed later in Section 2.3.3.

2.3.2 *Angular Momentum and Moments of Inertia.*

Angular momentum for a rigid body is derived following the same principals as those used in the preceding center of mass discussion. By following Wiesel and Halfman's work, along with the previous notation, the basic equations for angular momentum and moments of inertia for a rotating rigid body are presented below (Wiesel, 1997:102-111; Halfman, 1962:202-237). The specifics of how these equations are applied to this research are detailed in Chapters III and results are presented in Chapter IV.

Wiesel states, "A rigid body consists of a large number of individual particles. The total angular momentum will be the sum of the angular momentum for each mass element…"(Wiesel, 1997:98). This can be written

$$\mathbf{H} = \int_{\mathbf{body}} \mathbf{r} \times \mathbf{v} dm \tag{7}$$

where **H** is the angular momentum and **v** is a time derivative of the position vector resulting in a velocity of the mass element as it moves about the center of mass. It would be beneficial to take the origin as a point in the rigid body and tie the reference frame to the body as in Figure 4. This figure was originally found in Wiesel, but was modified to correspond to the notation system used in this thesis (Wiesel, 1997:102).

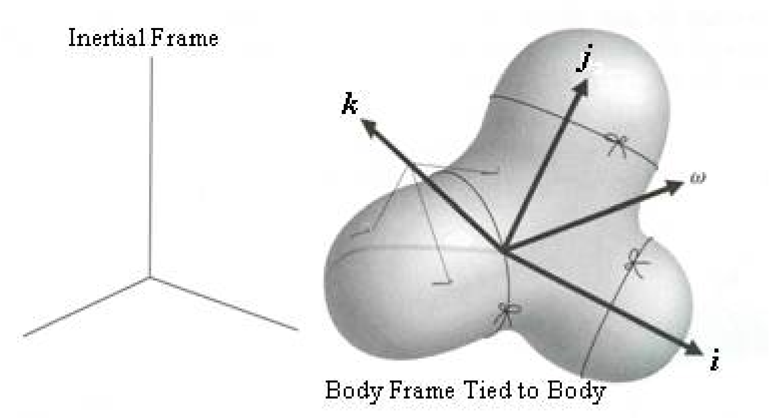


Figure 4. Reference Frame Tied to the Body and in an Inertial Frame (Wiesel, 1997:102)

This enables the velocity of any point in the rigid body to be described by only the rotation of the body as

$$\mathbf{v} = \vec{\omega} \times \mathbf{r} \tag{8}$$

where $\vec{\omega}$ is the angular velocity of the rigid body. The total angular momentum of a rigid body about an origin in a body frame of reference is then

$$\mathbf{H} = \int_{body} \mathbf{r} \times \left(\vec{\omega} \times \mathbf{r} \right) dm \tag{9}$$

While Equation 9 is a usable expression to determine the angular momentum, it would be better to expand **r** and $\vec{\omega}$ into their XYZ components in relation to the body frame. This is shown in Figure 4 above and is accomplished by writing **H**, **r**, and $\vec{\omega}$, using the unit vectors $\hat{\boldsymbol{i}}$, $\hat{\boldsymbol{j}}$, and $\hat{\boldsymbol{k}}$ yielding

$$\mathbf{r} = x\hat{\boldsymbol{i}} + y\hat{\boldsymbol{j}} + z\hat{\boldsymbol{k}} \tag{10}$$

$$\vec{\omega} = \omega_x \hat{\boldsymbol{i}} + \omega_y \hat{\boldsymbol{j}} + \omega_z \hat{\boldsymbol{k}} \tag{11}$$

Then using the vector identity $\mathbf{A} \times \left(\mathbf{B} \times \mathbf{C} \right) = \mathbf{B}\left(\mathbf{A} \cdot \mathbf{C} \right)$ **-** $\mathbf{C}\left(\mathbf{A} \cdot \mathbf{B} \right)$ produces the following angular momentum component equations

$$\mathbf{H}_x = \omega_x \int_{body} \left(y^2 + z^2 \right) dm - \omega_y \int_{body} yxdm - \omega_z \int_{body} zxdm \tag{12a}$$

$$\mathbf{H}_y = -\omega_x \int_{body} xydm + \omega_y \int_{body} \left(x^2 + z^2 \right) dm - \omega_z \int_{body} zydm \tag{12b}$$

$$\mathbf{H}_z = -\omega_x \int_{body} xzdm + \omega_y \int_{body} yzdm - \omega_z \int_{body} \left(x^2 + y^2 \right) dm \tag{12c}$$

The integrals above are in the form of the second moments of mass for an axis system attached to the rigid body and will not change in value unless mass is added or removed from the body. These integrals are called MOI and products of inertia; they both contain all of the information regarding the mass distribution within the rigid body in question. As Baruh points out

> Just as the mass of a body represents its resistance to translational motion, the distribution of the mass about a certain axis represents the body's resistance to rotational motion about that axis (Baruh, 1999:323).

Therefore, MOI define how a mass is distributed with respect to an axis and are defined as

$$\mathbf{I}_{xx} = \int_{body} \left(y^2 + z^2\right) dm \tag{13a}$$

$$\mathbf{I}_{yy} = \int_{body} \left(x^2 + z^2\right) dm \tag{13b}$$

$$\mathbf{I}_{zz} = \int_{body} \left(x^2 + y^2\right) dm \tag{13c}$$

The products of inertia define how a mass is distributed with respect to a plane and are given by

$$\mathbf{I}_{xy} = \mathbf{I}_{yx} = \int_{body} xydm \tag{14a}$$

$$\mathbf{I}_{yz} = \mathbf{I}_{zy} = \int_{body} yzdm \tag{14b}$$

$$\mathbf{I}_{xz} = \mathbf{I}_{zx} = \int_{body} xzdm \tag{14c}$$

Figure 5 below illustrates how the MOI for $\mathbf{I}_{xx}$ is developed and the remaining MOI and products of inertia are develop similarly. This figure from Baruh is modified to correspond to the notation used within this thesis (Baruh, 1999:327).

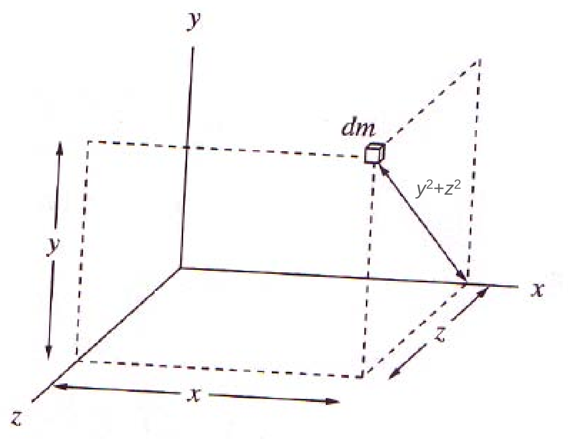


Figure 5. Illustration for Determining MOI about the X Axis (Baruh, 1999:327)

Combining the moments and products of inertia into a matrix form yields the commonly referred to moment of inertia matrix

$$\mathbf{I} = \begin{pmatrix} \mathbf{I}_{xx} & -\mathbf{I}_{xy} & -\mathbf{I}_{xz} \\ -\mathbf{I}_{yx} & \mathbf{I}_{yy} & -\mathbf{I}_{yz} \\ -\mathbf{I}_{zx} & -\mathbf{I}_{zy} & \mathbf{I}_{zz} \end{pmatrix} \tag{15}$$

The angular momentum is therefore related linearly through the MOI matrix and angular velocity

$$\begin{pmatrix} \mathbf{H}_x \\ \mathbf{H}_y \\ \mathbf{H}_z \end{pmatrix} = \begin{pmatrix} \mathbf{I}_{xx} & -\mathbf{I}_{xy} & -\mathbf{I}_{xz} \\ -\mathbf{I}_{yx} & \mathbf{I}_{yy} & -\mathbf{I}_{yz} \\ -\mathbf{I}_{zx} & -\mathbf{I}_{zy} & \mathbf{I}_{zz} \end{pmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \tag{16}$$

or in a concise matrix vector product

$$\mathbf{H} = \mathbf{I}\vec{\omega} \tag{17}$$

This research effort will exploit Equation 17 through the use of conservation of angular momentum to determine the MOI for a rigid body and therefore determine the change in fuel for a satellite. The theory for how Equation 17 will be used is outlined in Section 2.6. While the precise method used to determine the MOI for SimSat in this research and also the change in satellite fuel is found in Section 3.3 and 3.5 respectively.

2.3.3 *Euler's Equations of Motion.*

Utilizing angular momentum and the previously developed center-of-mass concept, the equations of motion for a rigid body can be developed. One must remember that when the sum of all torques acting on a system is zero, the total angular momentum of a system is conserved. Therefore, the total angular momentum of a torque free system is constant. This driving principle allows for the development of Euler's Equation of Motion for a rigid body that follows and is the basis for the mathematical models used in this research.

Rearranging Equation 1 and taking the cross product of both sides of the equation with some position vector **r** (which describes the location for the applied force) results in the applied moment, or torque, acting on the body being equal to the rate of change of the angular momentum. This follows directly from Wiesel.

$$\mathbf{M} = \left(\frac{d\mathbf{H}}{dt} \right)_{inertial\_frame} \tag{18}$$

Due to the choice of the body frame of reference for the previous development of Equation 17, the use of the following identity is required in order to obtain the time derivative of angular momentum in the inertial frame

$$\left(\frac{d\mathbf{A}}{dt}\right)_{inertial\_frame} = \left(\frac{d\mathbf{A}}{dt}\right)_{body\_frame} + \vec{\omega} \times \mathbf{A} \tag{19}$$

Equation 19 shows the time derivative of a vector **A** with respect to an inertial frame is equal to the time derivative of vector **A** in the body frame and the angular velocity cross product. Applying Equation 19 to Equation 17 yields

$$\mathbf{M} = \mathbf{I}\dot{\omega} + \vec{\omega} \times \mathbf{I}\vec{\omega} \tag{20}$$

where $\dot{\omega}$ is the time derivative of the body frame angular velocity, or acceleration, of the rigid body.

In order to simplify the rotational equations of motion found in Equation 20, an appropriate axis frame must be selected. If the center of mass for a rotating body is chosen as the origin for the axis system and the axes are aligned with the principle axes of the rigid body, the products of inertia, Equations 14a, b, and c, simplify to

$$\mathbf{I}_{xy} = \mathbf{I}_{yx} = 0 \tag{21a}$$

$$\mathbf{I}_{yz} = \mathbf{I}_{zy} = 0 \tag{21b}$$

$$\mathbf{I}_{xz} = \mathbf{I}_{zx} = 0 \tag{21c}$$

due to the rigid body being symmetric about the selected center of mass axis frame. This special case is called the principal axis frame and results in the moment of inertia matrix becoming

$$\mathbf{I} = \begin{pmatrix} \mathbf{I}_{xx} & 0 & 0 \\ 0 & \mathbf{I}_{yy} & 0 \\ 0 & 0 & \mathbf{I}_{zz} \end{pmatrix} \tag{22}$$

Using the principle axis frame of reference, the angular velocity and applied moment are resolved into the XYZ coordinates of the body frame. This process converts Equation 20 into three coupled, nonlinear differential equations known as Euler's Equations of Motion for a rigid body (Wiesel, 1997:111). The equations are listed below

$$\mathbf{M}_x = \mathbf{I}_{xx}\dot{\omega}_x + \left(\mathbf{I}_{zz} - \mathbf{I}_{yy}\right)\omega_y\omega_z \tag{23a}$$

$$\mathbf{M}_y = \mathbf{I}_{yy}\dot{\omega}_y + \left(\mathbf{I}_{xx} - \mathbf{I}_{zz}\right)\omega_x\omega_z \tag{23b}$$

$$\mathbf{M}_z = \mathbf{I}_{zz}\dot{\omega}_z + \left(\mathbf{I}_{yy} - \mathbf{I}_{xx}\right)\omega_x\omega_y \tag{23c}$$

2.3.4 *Orienting a Satellite System.*

Euler's Equations of Motion for a rigid body found in Equations 23a, b, and c provide a means to understand how an applied torque (**M**) will relate to a rigid body's MOI, angular velocity, and angular acceleration. Assume the rigid body is a non-rotating satellite orbiting the earth. The satellite is operating in a torque free environment where the $\sum \mathbf{M} = 0$. In order to control and orient the satellite, a reaction wheel consisting of a large flywheel attached to a motor is mounted rigidly within the satellite. The satellite requires at least three reaction wheels as a condition to obtain motion in all three axes. For simplicity, assume these wheels are oriented along the principle body axes of the satellite. When the reaction wheels spin about their axis and create a torque, the satellite rotation is of an equal and opposite torque in accordance with conservation of angular momentum. Figure 6 illustrates this concept showing a reaction wheel with a MOI of I and spinning about the **b**$_1$ axis with a velocity of ω_f. The satellite rotates about the **b**$_1$ axis in the opposite direction at a velocity of ω_1 resulting in a total angular momentum of zero based in some inertial frame of reference (Wiesel, 1997:142-143).

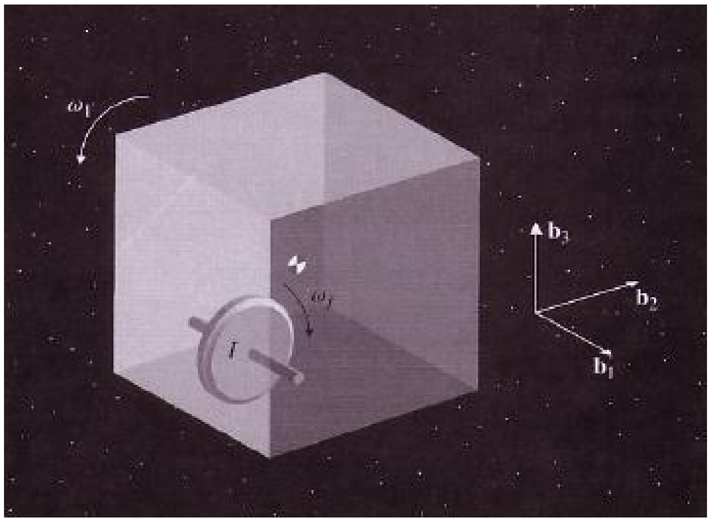


Figure 6. Spacecraft with a Reaction Wheel (Wiesel, 1997:143)

Using Equation 20 to develop the torque generated by the reaction wheel attached to the satellite rigid body as it appears in the satellite body frame of reference yields

$$\mathbf{M}_i = \mathbf{I}_{rw}\left(\dot{\omega}_{rw_i} + \dot{\omega}_{sat_i}\right) \tag{24}$$

where *i* corresponds to the *x*, *y*, or *z* axis. $\mathbf{I}_{rw}$ is the MOI for the reaction wheel as it spins about its axis, $\dot{\omega}_{rw}$ is the reaction wheel acceleration, and $\dot{\omega}_{sat}$ is the satellite acceleration. Since the satellite is operating in a torque free environment, Equation 24 results in the reaction wheel acting as an external torque on the satellite. Setting Equation 23 and 24 equal to each other for the same axis and remembering that the sum of the moments about an axis must equal zero due to conservation of momentum, the equations for each axis from some inertial frame of reference become

$$-\mathbf{I}_{rw_x}\left(\dot{\omega}_{rw_x}+\dot{\omega}_{sat_x}\right)=\mathbf{I}_{sat_{xx}}\dot{\omega}_{sat_x}+\left(\mathbf{I}_{sat_{zz}}-\mathbf{I}_{sat_{yy}}\right)\omega_{sat_y}\omega_{sat_z} \tag{25a}$$

$$-\mathbf{I}_{rw_y}\left(\dot{\omega}_{rw_y}+\dot{\omega}_{sat_y}\right)=\mathbf{I}_{sat_{yy}}\dot{\omega}_{sat_y}+\left(\mathbf{I}_{sat_{xx}}-\mathbf{I}_{sat_{zz}}\right)\omega_{sat_x}\omega_{sat_z} \tag{25b}$$

$$-\mathbf{I}_{rw_z}\left(\dot{\omega}_{rw_z}+\dot{\omega}_{sat_z}\right)=\mathbf{I}_{sat_{zz}}\dot{\omega}_{sat_z}+\left(\mathbf{I}_{sat_{yy}}-\mathbf{I}_{sat_{xx}}\right)\omega_{sat_x}\omega_{sat_y} \tag{25c}$$

where $\mathbf{I}_{sat}$ is the MOI of the satellite about the specified XYZ axis. Solving the above equations for the acceleration of the satellite yields the rigid body dynamics of the satellite rigid body from some inertial frame of reference as they are represented by SimSat's experimental hardware

$$\dot{\omega}_{sat_x}=\frac{-\mathbf{I}_{rw_x}\dot{\omega}_{rw_x}}{\mathbf{I}_{sat_{xx}}+\mathbf{I}_{rw_x}}+\frac{\left(\mathbf{I}_{sat_{yy}}-\mathbf{I}_{sat_{zz}}\right)\omega_{sat_y}\omega_{sat_z}}{\mathbf{I}_{sat_{xx}}+\mathbf{I}_{rw_x}} \tag{26a}$$

$$\dot{\omega}_{sat_y}=\frac{-\mathbf{I}_{rw_y}\dot{\omega}_{rw_y}}{\mathbf{I}_{sat_{yy}}+\mathbf{I}_{rw_y}}+\frac{\left(\mathbf{I}_{sat_{zz}}-\mathbf{I}_{sat_{xx}}\right)\omega_{sat_x}\omega_{sat_z}}{\mathbf{I}_{sat_{yy}}+\mathbf{I}_{rw_y}} \tag{26b}$$

$$\dot{\omega}_{sat_z} = \frac{-\mathbf{I}_{rw_z}\dot{\omega}_{rw_z}}{\mathbf{I}_{sat_{zz}}+\mathbf{I}_{rw_z}} + \frac{\left(\mathbf{I}_{sat_{xx}} - \mathbf{I}_{sat_{yy}}\right)\omega_{sat_x}\omega_{sat_y}}{\mathbf{I}_{sat_{zz}}+\mathbf{I}_{rw_z}} \tag{26c}$$

Equations 26a, b, and c show that the acceleration or angular rate of SimSat is a function of the MOI of SimSat, MOI of the reaction wheel, reaction wheel acceleration, and angular velocity of SimSat. These equations are the basis for the mathematical model used in this research; the specific application methodology is presented in Section 3.4.

2.4 Dynamic Response Analysis

Previous work by Dabrowski had only considered small orientation angle changes. Therefore, in order to ensure the capability of simulating large changes in orientation angle maneuvers via SimSat, the response of the reaction wheels needs to be characterized. This dynamic response analysis was accomplished through imparting a step command on one of SimSat's reaction wheels and analyzing both the reaction wheel's and SimSat's response. This section reviews the equations and theory used to determine the transfer functions for these systems.

By comparing Matlab® plots of the commanded response to the actual response of a system, it can be determined what the relationship is between the input and output of the commanded signal. A plot of actual system output response versus time is similar to Figure 7 below. Applying the following equations from Ogata

$$t_r \cong \frac{1.8}{\omega_n} \tag{27}$$

$$t_p = \frac{\pi}{\omega_d} \tag{28}$$

$$M_p = e^{\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}} \tag{29}$$

$$\omega_d = \omega_n \sqrt{1-\zeta^2} \tag{30}$$

where t_r, rise time, is the amount of time it takes to approximately reach the required response, ω_n, is the undamped natural frequency of the system, t_p, peak time, is the time it takes to reach the first peak of overshoot, ω_d, is the damped natural frequency of the system, M_p, maximum percent overshoot, is the maximum amount the response overshoots its final value divided by its final value multiplied by one hundred, and ζ, is the damping ratio of the system (Ogata, 2002:230-235).

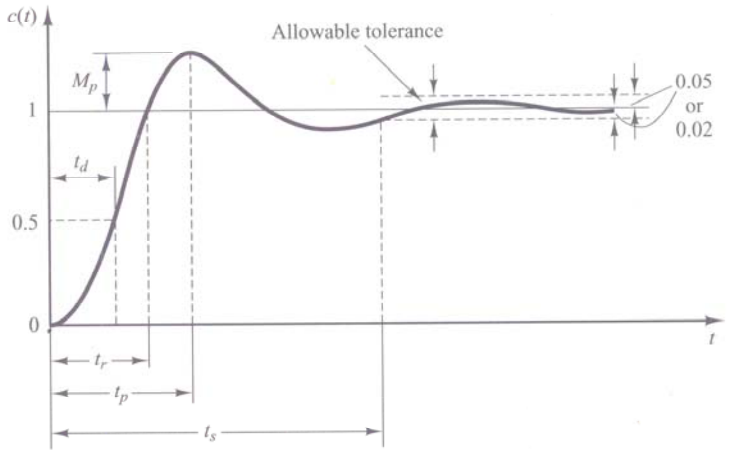


Figure 7. Unit Step Response Curve Showing t_d, t_r, t_p, M_p, and t_s (Ogata, 2002:230)

Through the use of Equations 27 – 30, and a plot of a systems output response, it is possible to find a second order transfer function that approximately represents the system in question directly from these parameters (Ogata, 2002:224-239; Franklin, Powell, Emami-Naeini, 1994:118-130). The transfer function will be in the form of Equation 31 below

$$G_{tf}\left(s\right)=\frac{{\omega_n}^2}{s^2+2\left(\zeta\right)\left(\omega_n\right)s+{\omega_n}^2} \tag{31}$$

Once the transfer function of the system is determined, the ordinary differential equations representing the system's characteristics can be attained. This is done by taking the inverse Laplace transform of the input and output equations to obtain the overall system differential equation. Application of this dynamic response analysis overview will be presented in Section 3.4 and the results in Section 4.4 and 4.5 as a means to determine the differential equations for SimSat and SimSat's reaction wheels.

2.5 Proportional Derivative Control

During the characterization of the reaction wheels and SimSat, the response of both systems was similar to a second order system. The reaction wheel system uses an Animatics SmartMotor™, which operates in a closed-loop fashion using a contractor proprietary control algorithm. The reaction wheel response can be mathematically modeled as a PD-loop. Section 3.2.2.1 discusses the reaction wheel system in further detail. Additionally, SimSat's response in a closed-loop control scheme is controlled through the use of a PD controller.

A PD controller, also known as a high pass filter, is designed to remove error found in a systems measured output. PD controllers accomplish this by subtracting the measured signal output from the desired output and determining the error in the signal. Mathematically a PD controller's transfer function is defined in the Laplace domain as

$$G_c(s) = K_p\left(1 + T_d s\right) \tag{32}$$

where K_p is the proportional gain constant designed to handle the steady-state requirement, and T_d is the time derivative constant used to control the speed of response of the system to an input. While Equation 32 is a PD controller's transfer function, Equation 33 below presents a more useable form of the transfer function when creating mathematical models and is in the form

$$G_c(s) = K_p + K_d s \tag{33}$$

where K_d is the result of multiplying K_p and T_d together. An illustration of how a PD controller is implemented in Simulink® is found in Figure 8 below.

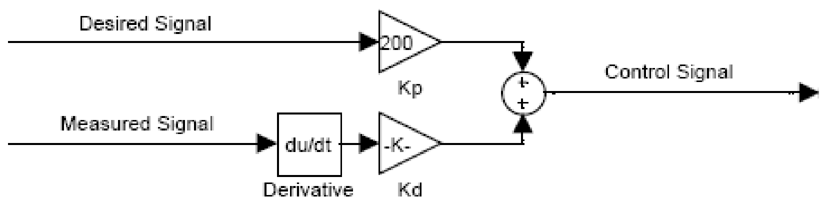


Figure 8. PD Controller Implemented in Simulink®

The differential equation in the time domain for both Equation 32 and 33 is found by taking the inverse Laplace of the transfer function and is found to be

$$g(t) = K_p\left(e + T_d \dot{e}\right) \tag{34}$$

where e is the error in the signal and $\dot{e}$ is the time derivative of the error in the signal (Ogata 2002:281-287). Section 3.4 details the values used for the reaction wheel and SimSat PD controllers used in this research.

2.6 Determining MOI for SimSat

In order to determine the initial values for the PD controller used in the mathematical models of SimSat, the MOI are required. The MOI for SimSat can be found through the use of conservation of angular momentum, Equation 17, and the open-loop response of SimSat to an applied torque. The theory for how the MOI can be determined from this information is presented below.

Since SimSat is considered a rigid body operating in a torque free environment when a reaction wheel spins about its axis, a torque is created and due to conservation of angular momentum SimSat must spin in an equal and opposite direction. Using this theory there are several ways to determine the MOI of SimSat based on data obtained from the hardware. The first method is to change the angular speed of the flywheel, and then use the mean constant velocity of the reaction wheel and SimSat, as shown in Figure 9, to determine the MOI of SimSat. Figure 9 shows the measured velocity data from SimSat as recorded by the onboard gyroscope for ω_{sat}, and the corresponding velocity encoders for ω_{rw}. Further description of the instrumentation is briefly described in Section 3.2.4 or detailed in Dabrowski's thesis (Dabrowski, 2001).

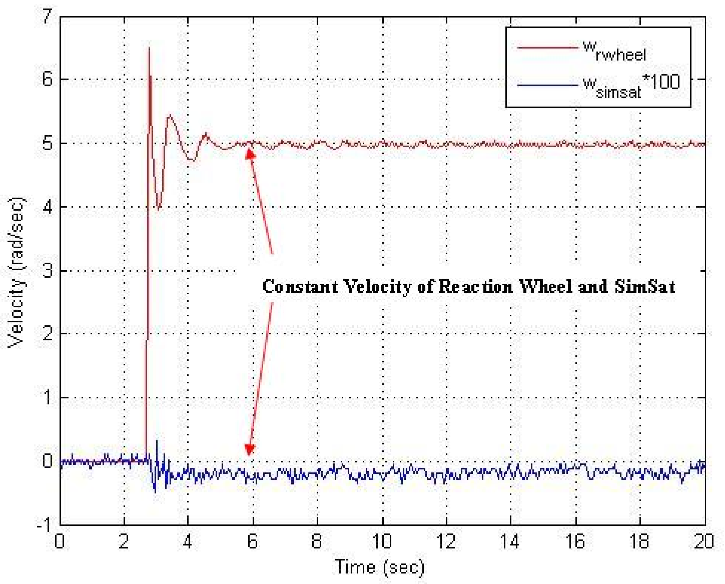


Figure 9. Reaction Wheel Velocity vs SimSat Velocity

SimSat's MOI is found using Equation 17 and taking the absolute values of the constant reaction wheel velocity and constant SimSat velocity, resulting in the equation below

$$\mathbf{I}_{sat} = \frac{\mathbf{I}_{rw}\left(\omega_{rw} - \omega_{sat}\right)}{\omega_{sat}} \tag{35}$$

where ω_{rw} is the angular velocity of the reaction wheel and ω_{sat} is the angular velocity of SimSat to determine SimSat's MOI (Wiesel, 1997:143).

The second method to determine the MOI of SimSat is to use the change in reaction wheel velocity and SimSat velocity and using the ratio between the two values at the peak point in the data to determine the MOI of SimSat. As seen in Figure 10 below it is extremely difficult to use this method for determining the MOI of SimSat based on the noise in the data. The data presented in Figure 10 has been smoothed using a running average technique.

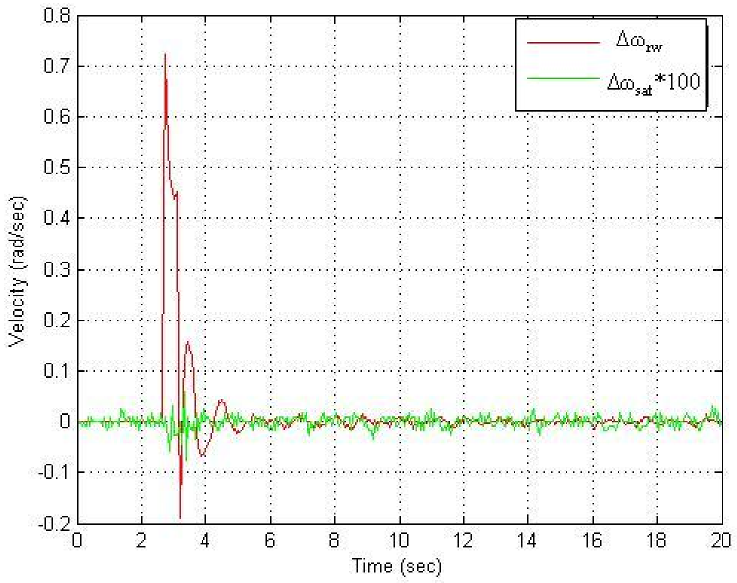


Figure 10. Reaction Wheel Change in Velocity vs SimSat Change in Velocity

The final theory used to determine the MOI of SimSat is to use the change in position of the reaction wheel and the change in position of SimSat. Once the two running positions are plotted, like in Figure 11, the slope of each of the lines can be determined.

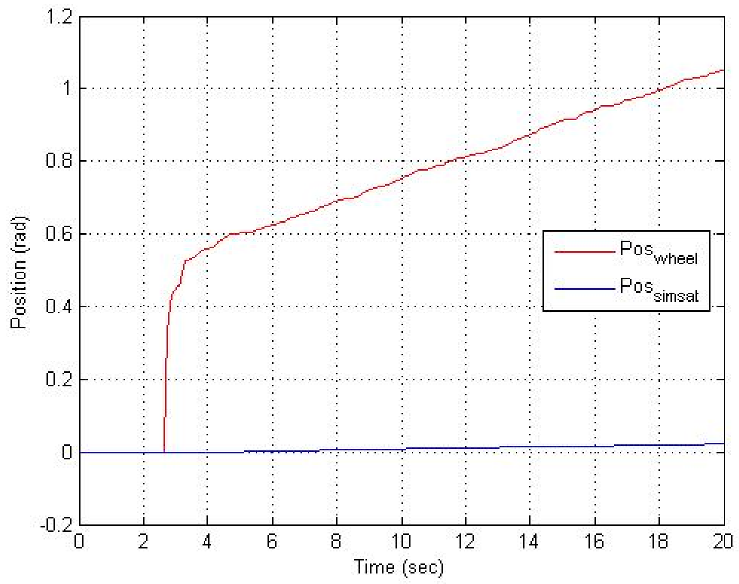


Figure 11. Running Position of Reaction Wheel vs Running Position of SimSat

The resulting ratio from the slopes can be used to determine $\mathbf{I}_{sat}$ by the following equation

$$\mathbf{I}_{sat} = \frac{\mathbf{I}_{rw}\left(slope_{rw} - slope_{sat}\right)}{slope_{sat}} \tag{36}$$

Equation 35 and 36 are related since the slope of the lines have the same units of velocity as ω_{rw} and ω_{sat} from Equation 35. However, Equation 36 is more useful to determine the MOI in the pitch and roll axis due to the SimSat's configuration and the measurements available. Since pitch and roll can only sustain small constant velocities imparted by the reaction wheel before gravitational effects impact SimSat's motion Equation 36 is better suited to determine the MOI in these directions.

Once the baseline MOI of SimSat are determined the PD control settings required in the mathematical models can be determined. The specific methods used in this research to determine the MOI of SimSat is presented in Section 3.3 and the results are presented in Section 4.3.

2.7 Least Squares Optimization

While the preceding open-loop methods to determine the MOI of a satellite are viable they are not very practical in an operational environment. An operational satellite will always be under some type of control and resulting motion to maintain its desired orientation. If the open-loop methods are constantly used on an operational satellite, it will always have to be repositioned to the desired orientation limiting its usefulness. Therefore, the ability to detect the change in MOI from any satellite motion offers the satellite operator a transparent way to determine MOI and thus the remaining satellite fuel.

The ideal scenario for space operations using the methods outlined in this research would be to determine the baseline MOI using open-loop methods when the satellite first reaches orbit. Then using the obtained MOI data, modify the mathematical model of the satellite until it corresponds to the actual satellite movements. Once the model and actual systems match, measured telemetry data from the satellite can be used by the model to determine the change in MOI and the resulting change in satellite fuel. See Figure 12 below for a flowchart of this on orbit process.

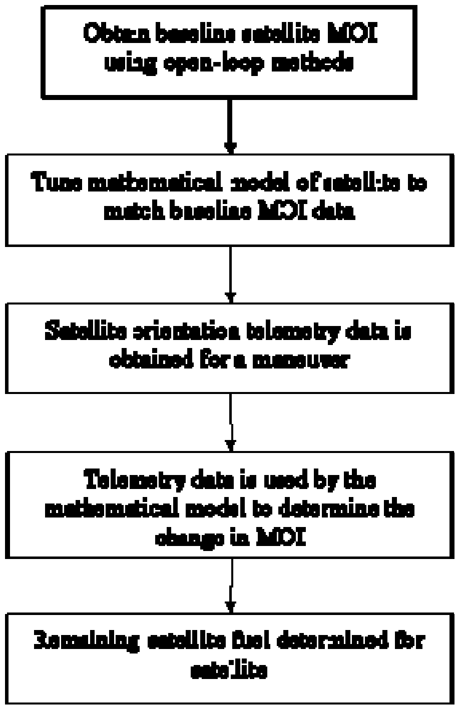

Obtain baseline satellite MOI using open-loop methods
Tune mathematical model of satellite to match baseline MOI data
Satellite orientation telemetry data is obtained for a maneuver
Telemetry data is used by the mathematical model to determine the change in MOI
Remaining satellite fuel determined for satellite

Figure 12. On Orbit Process to Determine Change in Satellite Fuel

The ability to determine the change in MOI from an initial condition to a present condition relies on a mathematical model of the system. Therefore, a math model was created as part of this research to accurately represent SimSat; and any changes made to the math model can be validated via SimSat and vice versa. The ability to compare the change in MOI, and subsequently the satellite fuel, is accomplished through the use of a least squares type optimization method.

An optimization problem "involves finding the values of *p* parameters $y_1,...,y_p$ that minimize a performance index that is a function of these parameters, $L(y_1,...,y_p)$" (Bryson, 1999:1). The performance index *L* mentioned by Bryson is often referred to as the cost function for the optimization problem. The cost function is dependent on the design parameters used in the problem and for this research is in the form below

$$J(p) = \sum_{i=1}^{n} \left| \overline{\theta}_i - \theta_i\left(p\right) \right|^2 \qquad (37)$$

where *p* is the design parameter, $\overline{\theta}_i$ is the measured hardware orientation data at a specific point, and θ_i is the modeled orientation data corresponding to the same point. The orientation data for $\overline{\theta}_i$ and θ_i can be from the yaw, pitch, or roll direction but the two sets of data used in the cost function must be from the same direction. The cost function in Equation 37 allows the design parameters to be changed in order minimize or maximize the function depending on what is desired.

An effective way to accomplish a least squares optimization is through the use of Matlab®'s Optimization Toolbox and the *fminsearch* function (Matlab, 2005). The *fminsearch* function is designed to find the minimum of the user defined cost function for a given function of several variables and some initial estimates for the variables. The function will then iterate on the given values/parameters and return the optimized values for the function once the cost function has been minimized. This type of optimization is called unconstrained nonlinear optimization.

The specific method used in this research to determine the change in satellite fuel along with further explanation of the specific cost function used is presented in Section 3.5. In Section 4.7 the results from the use of the least squares type optimization and the ability to detect the change in satellite fuel are presented.

2.8 Chapter Summary

This chapter started with a review of related research that has been conducted in the field of mass property estimation techniques for satellite systems. Next, the theoretical foundation for this research project was presented. Theoretical concepts included center of mass determination, angular momentum, MOI, Euler’s Equations of Motion, rigid body dynamics of SimSat, orienting a satellite system, dynamic response analysis, PD control, and an optimization technique to determine the change in satellite fuel. Specific methods to accomplish the research objectives are documented in Chapter III. Chapter IV discusses the results whereas Chapter V provides conclusions and recommendations for future efforts.

III. Methodology

3.1 Chapter Overview

The purpose of this chapter is to detail the specific equipment, models, and research approach used to characterize techniques for determining remaining satellite propellant using measured MOI. The methodology developed in this chapter relates how the principals and theories from Chapter II will be applied in support of this research effort. First, an explanation of the experimental hardware known as SimSat is presented. Then, the methods for determining the characteristics of the reaction wheels and SimSat are explained, followed by the mathematical models. Finally, the method used to determine the MOI and the resulting change in satellite propellant using optimization techniques are presented. The entire process, outlined briefly here, is described in depth in the sections that follow.

3.2 SimSat Hardware

SimSat was designed and constructed in 1999 as part of Colebank, Jones, Nagy, Pollak, and Mannebach's Systems Engineering Master's Thesis at AFIT. Their charter was to design a system for AFIT to "simulate satellite behavior with as much fidelity as possible" (Colebank et al, 1999:1-3). Their resulting design consists of three main parts: the pedestal containing an air bearing assembly (previously shown in Figure 2), the satellite architecture hardware (SimSat), and the ground station computer which provides real-time data transmission and control of SimSat. A detailed design specification and description of the original configuration can be found in the thesis of Colebank et al.

Over the past several years, SimSat has been modified and upgraded in response to several students' thesis objectives. A brief synopsis of the major changes to SimSat's baseline configuration is outlined as follows; for a more detailed description of the modifications please refer to each identified thesis. Dabrowski initiated the changes to SimSat's configuration with a major upgrade to SimSat's original reaction wheels along with converting SimSat from analog to mostly digital signals. Furthermore, Dabrowski upgraded the onboard computer and ground station hardware to take advantage of more stable software (Dabrowski, 2003). French installed and integrated a laboratory safe cold gas thruster control system (French, 2003). Kimsal installed a near-infrared and color video camera allowing SimSat to autonomously track a heat source (Kimsal, 2004). Smith upgraded the existing mechanical gyro with a new fiber optic gyro (Smith, 2005). These previous modifications to SimSat remain intact. Reference Sections 3.2.1 - 3.2.3 to review the current physical configuration as used in this research effort.

3.2.1 *Air Bearing System.*

SimSat achieves three DOF rotational motion through the use of a Space Electronics, Inc Model SE9791 Tri-Axis Spherical air bearing system. It accomplishes this freedom of motion by supporting an 8 in diameter spherical centerpiece on a 12.7 μm cushion of air. This air cushion is provided by a system of six jets, funneling compressed air at approximately 500 kPa into the bottom of the air bearing cup that is attached to the pedestal. The pedestal allows SimSat to achieve unrestricted rotational motion in two directions while limiting the third direction of rotational motion to ±25° due to contact of the aluminum attachment plates with the pedestal. The direction of rotation that is limited is defined by the coordinate system selected for use with SimSat. This research effort used the principle axis frame of reference with the origin of the coordinate system corresponding to the center of the spherical centerpiece. The resulting SimSat body fixed axes and corresponding MOI are defined in Figure 13.

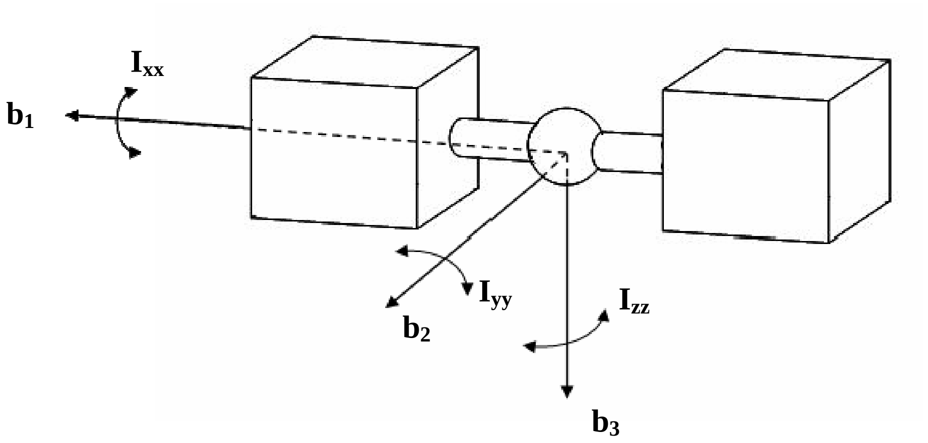


Figure 13. SimSat Body Fixed Axes and MOI Defined

With SimSat's axes as defined in Figure 13, the rotation about the b_2 axis, or "pitch", will be limited to ±25° from the balanced baseline condition. The resulting rotation about the b_1 axis will be called "roll" and rotation about the b_3 axis will be called "yaw". These notations shown in Figure 14 will be used throughout the remainder of this thesis to describe the motion of SimSat.

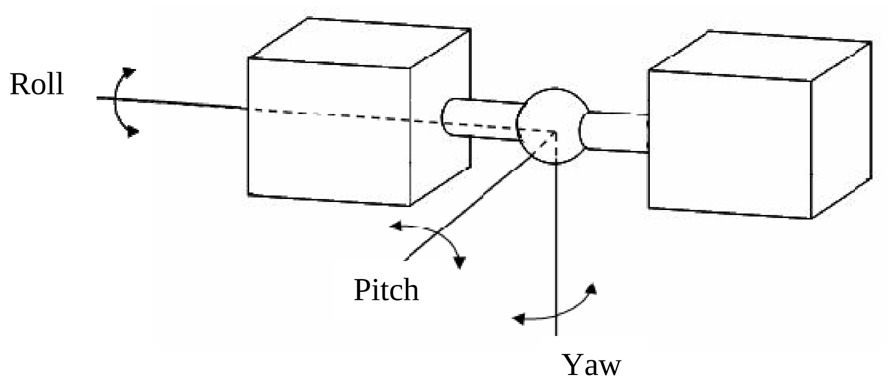


Figure 14. SimSat Orientation defined in Yaw, Pitch, and Roll

3.2.2 *Satellite Architecture.*

The satellite architecture hardware, or SimSat, that sits atop the air bearing assembly is comprised of several different systems which are mounted to aluminum plates connected together using stainless steel mounting rods and locking collars. SimSat's configuration is in the shape of a dumbbell with its overall physical dimensions measuring 72 x 21 x 14 inches with an approximate weight of 250 lbs. The major systems attached to the aluminum plates include: three batteries providing 12, 24, and 36 volts of power for SimSat's systems; a three axis fiber optic gyroscope for attitude determination; an onboard computer; a near-infrared and color video camera/transmitter; three independent reaction wheels; and a cold gas thruster system for attitude control. These systems are identified in Figure 15.

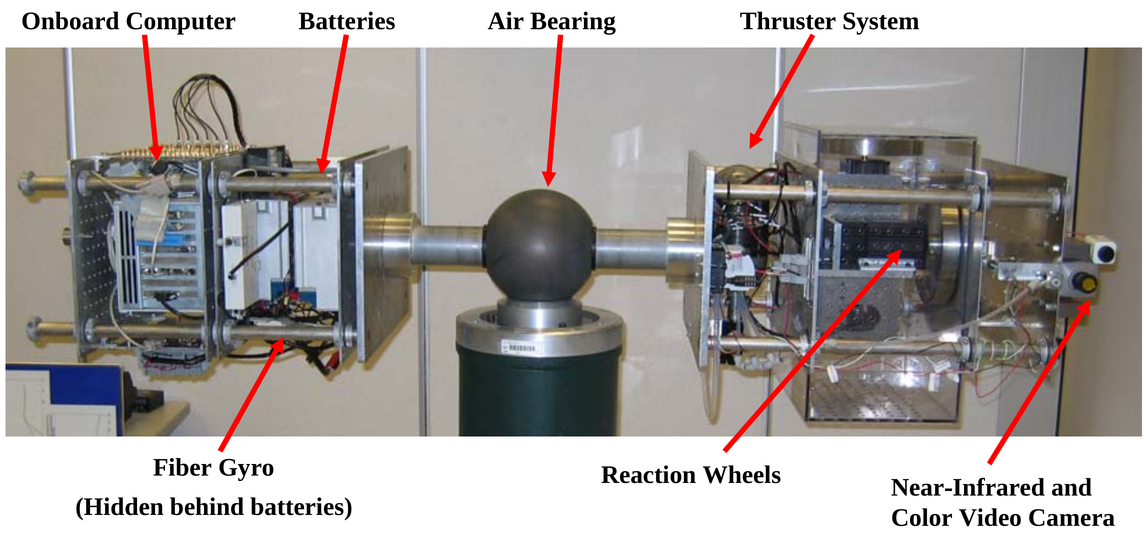


Figure 15. Major Systems for SimSat

This research predominantly used the reaction wheels, fiber optic gyroscope, power supply, and onboard computer. The details for these systems are presented in the rest of this section.

3.2.2.1 ***Reaction Wheels.***

The ability to orient and control SimSat's motion is provided through the use of three reaction wheels as shown in Figure 16. The reaction wheels were fabricated in the AFIT lab and consist of a steel rim attached to an 8.625 in diameter aluminum disk. The MOI for each of the reaction wheels has been calculated to be 1.955 x 10^{-2} $\text{kg} \cdot \text{m}^2$ (Dabrowski, 2003:3-6).

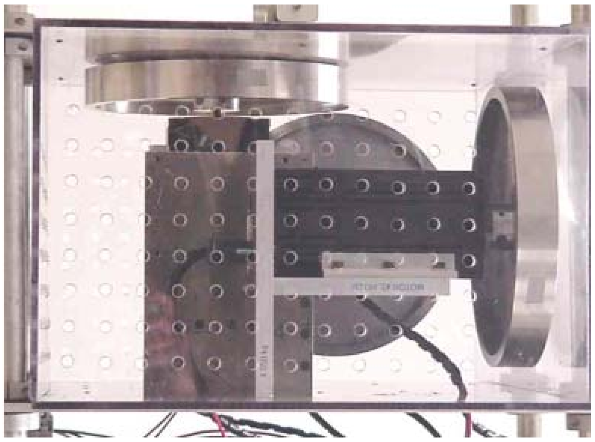


Figure 16. SimSat's Reaction Wheel Configuration

Each reaction wheel is independently controlled and driven by an Animatics SmartMotor™ Model SM3450 motor system. Each motor system integrates a brushless DC servo motor, motion controller, encoder, and amplifier into a single package (Dabrowski, 2003:3-5). The Animatics SmartMotor™ Model SM3450 characteristics are listed in Table 1.

Table 1. Animatics SmartMotor™ Model SM3450 Characteristics

Parameter	Value
Weight	2.90 kg
Length	155 mm
Width	82.6 mm
Voltage	36V
Encoder Resolution	4,000 cts/rev
Data Interface	RS232

The Animatics SmartMotor™ uses an internal closed-loop configurable proportional plus integral plus derivative (PID) controller with velocity and acceleration feed-forward control action combined with the motor, amplifier, and encoder into a single unit. The integrated motor settings used for this research are found in Table 2.

Table 2. Animatics SmartMotor™ Model SM3450 Control Settings Used

Parameter	Value
Proportional Coefficient	25
Integral Coefficient	0
Integral Limit	0
Differential Coefficient	4000
Velocity Feed Forward Coefficient	1000
Acceleration Feed Forward Coefficient	10,000
Acceleration Limit	250
Error Tolerance	32,000

Due to the proprietary nature of the PID controller and motor used, an exact mathematical model of the reaction wheel system is not available. However, in Dabrowski’s previous work he utilized a look-up table to characterize the acceleration of the reaction wheel based on a commanded change in wheel velocity (Dabrowski, 2003:2-13). This method was successful when using a ±1° detection maneuver as part of his research. The author attempted to use the look-up table as part of this research. However, after running several test cases it was apparent that the large changes in satellite orientation and wheel speeds needed for this research would not be captured. This resulted in a new mathematical model for the reaction wheel being created that would accurately represent the reaction wheels used in this research. The method for the creation of the reaction wheel math model is detailed in Section 3.4.1.

3.2.2.2 ***<u>Fiber Optic Gyroscope.</u>***

Prior to 2005 all testing utilizing SimSat relied on a Humphrey model CF-75-0201-1 axis rate gyroscope to determine the angular velocity and hence the position of SimSat. This mechanical helicopter gyroscope was identified as an error source by several authors as it suffered from temperature sensitivity resulting in a phenomenon known as gyro drift. As part of his research, Kimsal, documented this drift rate and determined

> As is evident, there is a distinct difference in the behavior of the gyro as it is allowed to warm up. Immediately upon start-up, the gyro exhibits a linear decay in reported angle. As time goes on, it appears to come to a limit; the 50 minute and 60 minute plots lie almost on top of one another (Kimsal, 2004:78-79).

As a result of this drift rate, a new fiber optic gyroscope was procured in 2002 and installed by Smith in 2005. The LN-200 model Fiber Optic Gyroscope, Figure 17, manufactured by Northrop Grumman® Navigation Systems is a space-qualified fiber optic gyroscope with up to a 1°/hr accuracy rating. The gyroscope's characteristics can be found in Table 3.

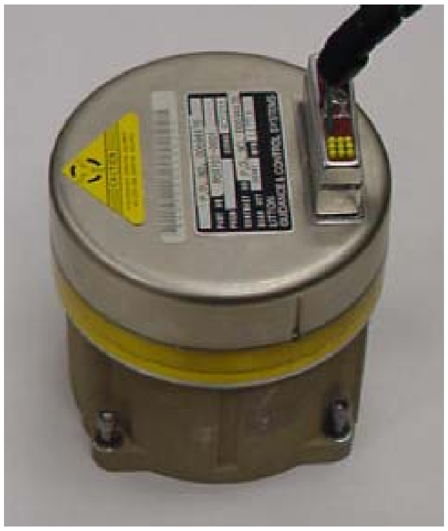


Figure 17. Fiber Optic Gyroscope Model LN-200

Table 3. Northrop Grumman® Fiber Optic Gyroscope Model LN-200 Characteristics

Parameter	Value
Weight	700 g
Diameter	8.9 cm
Height	8.5
Power Consumption	10 W
Bias Repeatability	1-10 / hr
Random Walk	0.04-0.1° $hr^{1/2}$ power spectral density
Data Latency	<1 msec
Data Protocol	RS-485
Data Structure	Synchronous Data Link Control (SDLC)

This improved gyroscope, installed in the same location where the previous gyro was located, was only partially tested by Smith.

> As of this writing the LN-200 gyroscope and interface board were installed but not yet fully integrated with SimSat…testing should be done to ensure that the fiber-optic gyroscopes do indeed mimic the output format of the original gyros (Smith, 2005:3-10)

Due to Smith's comments, the author researched the installation and resulting data from the fiber gyro. It was determined the telemetry data provided was intermittent. The author discovered the wiring connecting the fiber gyro to the interface board was not properly connected and promptly corrected the deficiency. To determine if the gyro was providing accurate readings in SimSat's yaw direction, a zero orientation point was marked on the floor and wall of the lab as shown in Figure 18. Additionally, a desired orientation point was marked on a piece of paper hung on the wall in the lab as in Figure 19.

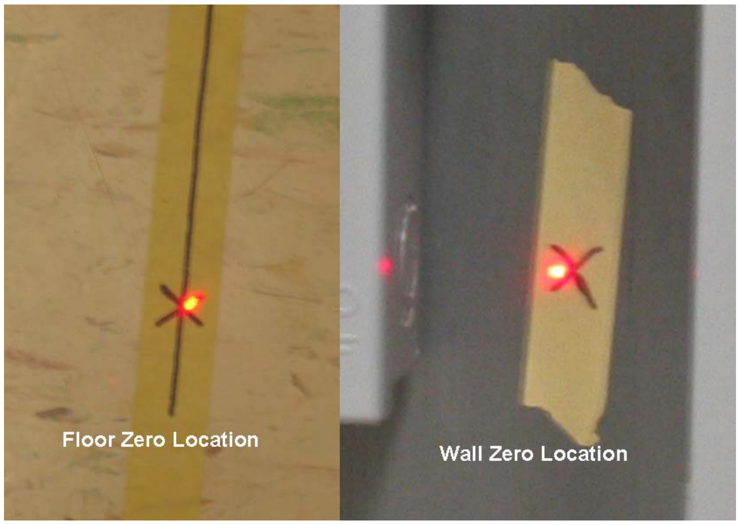


Figure 18. Zero Orientation Markings on Floor and Wall

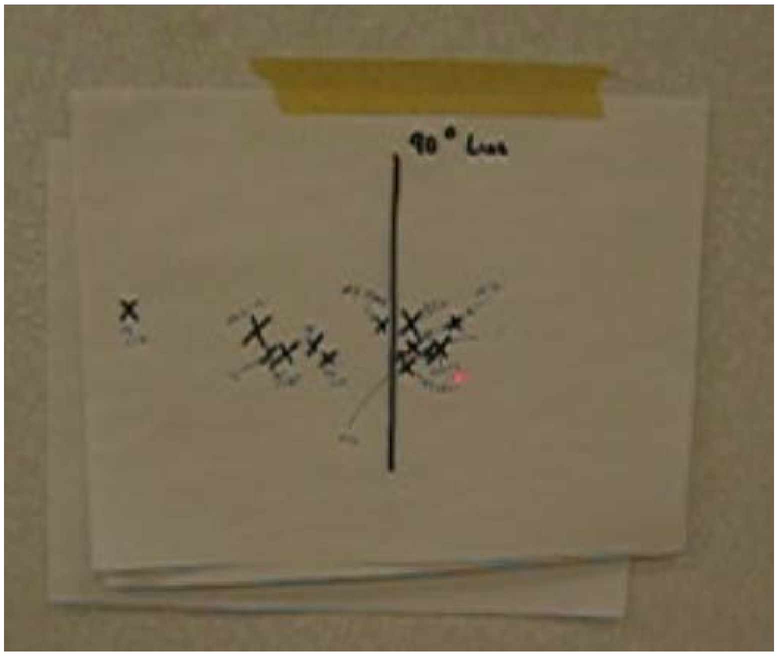


Figure 19. Desired Orientation with Initial and Final SimSat Orientation Marks

This desired orientation point was chosen as +90° since that required SimSat to make a significant orientation change and could easily be determined through geometric means in the lab. In order to determine if SimSat was achieving the desired orientation, two laser pointers were attached to SimSat and aligned with the zero orientation point markings as shown in Figure 18. The use of two laser pointers and markings ensured the yaw orientation was properly aligned and could be compared to actual telemetry data from the fiber gyro. Once SimSat was aligned with the zero orientation markings it was commanded to the desired orientation and the location where the laser pointer settled was marked on the paper as in Figure 19. SimSat was held at this desired orientation until SimSat suffered a wheel malfunction or the gyro suffered a data spike corrupting the data as shown in Figure 20. This final position, shown by the laser pointer before the loss of the telemetry data, was marked on the paper in order to calculate the drift associated with the gyroscope. The results from this experiment are found in Section 4.2.

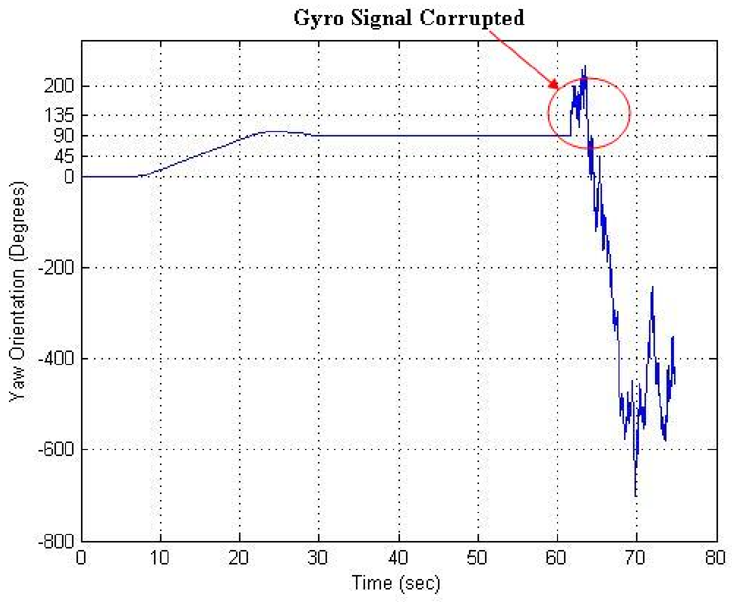


Figure 20. Corrupted Gyro Telemetry Data from SimSat

As mentioned previously a wheel malfunction occurred when the reaction wheel stopped responding to the commanded input, or a spike in the commanded reaction wheel speed made the reaction wheel accelerate to an undesirable speed. This anomaly was very infrequent and only required the data run to be stopped and started over. However, the fiber optic gyro was extremely susceptible to errors and at different times would give random and highly erroneous data as shown in Figure 20. This event would occur with no warning, and could happen during the middle of a data run or in between runs. When the gyro suffered an error the entire SimSat system and ground station had to be turned off and restarted. The frequency of the errors required the author to try several fault isolation techniques to determine the cause of the gyro data corruption. Unfortunately, a precise cause for the problem was not determined and the findings and recommendations for correcting the fiber gyro are found in Section 5.2.3.

3.2.2.3 ***Power Supply.***

Since SimSat is free to move in three degrees of freedom the power required for SimSat's systems comes from three Power-Sonic® Model PS-12180 rechargeable batteries. Each of the 12 V sealed lead-acid batteries has a rated capacity of 18 Amp Hours when discharged at the one hour rate (Power-Sonic, 2006). The wiring assembly on SimSat allows for the selection of 12 V, 24 V, and 36 V to power the different systems on SimSat. In order to fully operate all of SimSat's accessory hardware, the 36 V power setting is required. Three fully charged batteries will operate SimSat for approximately three hours depending on the use rate and commanded speeds for the reaction wheels.

3.2.2.4 ***Onboard Computer.***

SimSat achieves real time telemetry, command, and control of the systems through the use of an on-board computer. This computer system is manufactured by dSPACE® Inc., and consists of a dSPACE® AutoBox® DS400 providing the DC computing power required by SimSat. The AutoBox® DS400 is currently configured with a DS1005 PPC Processor Board which runs programs created, compiled, and uploaded to the board from SimSat's operator. Three RS-232 serial ports provide communication between the reaction wheels and the AutoBox® DS400. Another RS-232 serial port provides communication between the AutoBox® DS400 and the fiber gyro through a specially designed SkEyes Unlimited Corporation Model SK-PCB-0201 LN-200 interface board. This interface board translates the SDLC data packets from the fiber gyro into a usable RS-232 signal for the AutoBox® DS400 (Smith, 2005:3-9-3-10). Communication and transmission of real time telemetry from SimSat to the ground station computer is achieved through a RadioLAN® DockLINK™ Model 408-008 wireless transmitter at speeds up to 10 mega bits per second (Mbps).

3.2.3 *Ground Station Computer.*

SimSat's ground station computer is a Dell® 4500 model computer running a 2.26 GHz Intel Pentium® 4 processor with 256 MB of RAM. It also contains a RadioLAN® PCI CardLINK™ RMG-160 card that permits the computer to communicate with SimSat via the RadioLAN® DockLINK™ wireless transmitter at up to 10 Mbps. This ground station computer runs Microsoft Windows® 2000 Professional as an operating system. The software packages installed on the computer that permit the real time operation of

SimSat are provided as part of a dSPACE® software package and are: ControlDesk® Version 2.2.5, Matlab® Version 6.5, Simulink® Version 5, Signal Processing, and Real Time Workshop®. How these computer programs are implemented to control SimSat is provided in the following section.

3.2.4 *SimSat Hardware Control.*

SimSat is controlled in real time through the use of an experiment creation and management tool called ControlDesk ®. ControlDesk® is integrated with Matlab® and Simulink® which allows programs and models created in Matlab® and Simulink® to be compiled and uploaded to SimSat's DS1005 PPC processor board through the wireless transmitter. The process to control SimSat and collect real time experimental data is briefly outlined below.

First, SimSat's operator creates a Simulink® hardware model for SimSat, similar to the one in Figure 21. Second, a new experiment is created in ControlDesk® and the newly created model is added to the experiment. The third step compiles the model. Then, the Real Time Workshop® links the variables identified in the model with a PPC file that is uploaded to the DS1005 PPC processor board on SimSat. Once the file is uploaded, the operator can create an interface, or layout, in ControlDesk® similar to Figure 22 and link the model variables to displays located on the layout. These controls and displays provide SimSat's operator real time data monitoring, control, and data acquisition of any variables identified in the hardware model. This was the method used to create, control, and acquire data from SimSat as part of this research. All hardware models used as part of this research effort can be found in Appendix A.

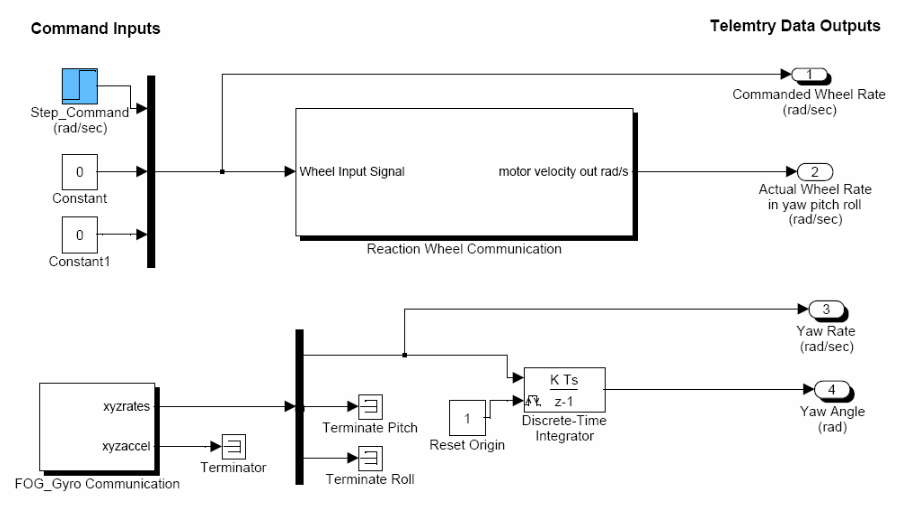


Figure 21. Simulink® Model used for Determining SimSat's Reaction Wheel Response

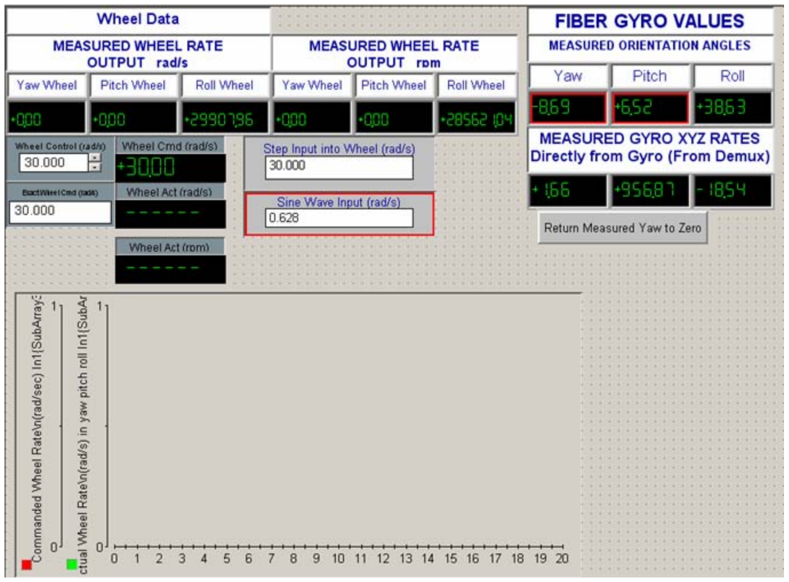


Figure 22. ControlDesk® Layout Used to Monitor SimSat Realtime and Capture Telemetry Data

3.3 Determining the Baseline MOI for SimSat

The math models being created in the following sections rely on baseline MOI information being obtained from the SimSat hardware. These baseline MOI values are used to calculate an initial estimate for the gain values used in the PD controller for SimSat, following the same approach as in Section 3.4.1 for the reaction wheel. The baseline MOI can also be used to verify the accuracy of the math models created in the following sections when compared to actual hardware data. The baseline MOI were found using the theory outlined in Section 2.6.

SimSat's yaw baseline MOI was found using the mean of the constant velocity of the reaction wheel and the resulting mean velocity of SimSat along with Equation 35. The yaw orientation and reaction wheel data used to determine the baseline MOI for SimSat are obtained from the reaction wheel response characterization methodology presented in Section 3.4.1. Using the *Determine_MOI_SimSat_3_Axis* Matlab® code, found in Appendix B, the MOI in the yaw direction was calculated for each of the reaction wheel settings.

The MOI for pitch and roll were not easily obtained. Due to gravitational effects in the pitch and roll axis (creating an additional moment on SimSat as it moves) the previous method used to determine the MOI in the yaw direction was not appropriate to calculate the MOI for these axes. Therefore, the use of Equation 36 using the change in position of both the reaction wheel and SimSat method was used for these two axes. This is due to the short duration that SimSat and the reaction wheel will both be at a constant velocity, and Equation 36 is the preferred equation based on the measurements available. To accomplish this method the reaction wheel settings for pitch and roll were changed in 5 rad/sec intervals. Then, using the same Matlab® code found in Appendix B, the change in position of the reaction wheel and SimSat were plotted and the resulting slope of each line was found. The results for calculating the MOI for SimSat in the yaw, pitch, and roll axes using the two methods in this section are found in Section 4.3.

3.4 Math Model Development

In order to determine if the change in satellite propellant can be ascertained from changes in MOI, mathematical models of SimSat's hardware were created. The math models are computer models designed to mimic the actual hardware system in use and allow for analysis of data generated by the hardware. All of the math models created in support of this research were generated using Simulink® Version 6.2 software and can be found in Appendix C. Simulink® is a model-based programming environment that takes mathematical equations and places them in blocks with inputs and outputs making it easier to create mathematical models of systems. Similarly, the use of Simulink® ensured versatility between the SimSat hardware and newly generated math models. However, down converting several math models to be used on the SimSat hardware proved to be difficult due to the differences in the versions of Simulink® software. The method for creating the reaction wheel math model and the various closed-loop math models for SimSat is presented in the following sections.

3.4.1 *Reaction Wheel Math Model.*

Since Dabrowski's math model for the reaction wheels was determined to be insufficient for this research effort as outlined in Section 3.2.2.1, it was necessary to create a new math model that would be more representative of the reaction wheel system. This method used the yaw reaction wheel on SimSat to determine the math model for each of the reaction wheels used on SimSat for several reasons. The selection of the yaw axis will substantially negate any gravity induced torques that would be generated by using the pitch or roll axes. Also, since all three reaction wheels are identical it is more beneficial to select the yaw wheel in order to capture SimSat's yaw orientation change data from a step command. This data can then be used to determine the MOI of SimSat as outlined in the previous section.

Using the theory for dynamic response analysis presented in Section 2.4, the transfer function and the resulting math model for the actual reaction wheel could be determined using the following method.

- Create a Simulink® model with a step command applied to SimSat's reaction wheel system and capture the actual reaction wheel speed from this command. Figure 21, previously presented in this chapter, shows the actual hardware model used in this effort.
- Change the input step command by 5 rad/sec until a satisfactory characterization of the wheel response to a step command was achieved. This occurred at 80 rad/sec due to the rate limiting effects on the reaction wheel being observed. The author determined the max rate limit had been reached due to the consistent response of the reaction wheel for a 60 - 80 rad/sec step command. The full results and analysis of the reaction wheel response to a step input is presented in Section 4.4.

- Plots of the commanded verses the actual reaction wheel speeds were produced by the Matlab® code found in Appendix D. The resulting plots, similar to the one in Figure 23 below can be found in Appendix E.

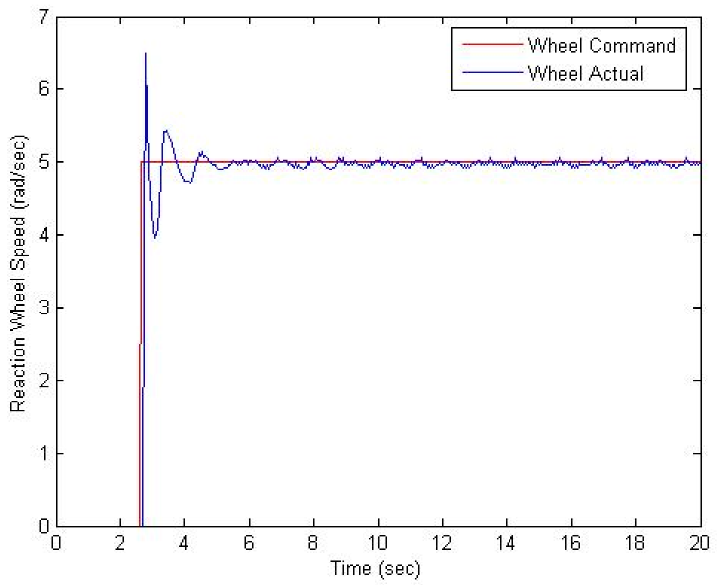


Figure 23. Plot of SimSat's Reaction Wheel Response Following a Commanded Step Input

- Through the use of Equations 27 – 30 and plots like Figure 23, the second-order transfer function describing the Laplace transform of the output response (the actual reaction wheel speed) to the Laplace transform of the input signal (commanded wheel speed) was obtained as

$$G_{rw}(s) = \frac{61.967}{s^2 + 5.626s + 61.967} \tag{38}$$

- Once Equation 38 was obtained, a math model for the reaction wheel could be created. This model is designed to take the commanded reaction wheel speed and provide the true reaction wheel speed observed from SimSat. The math model accomplishes this conversion by using a PD controller, as described in Section 2.5, in order to generate the second-order response that is observed in the data. This also corresponds with the true hardware settings as outlined in Table 2, which show the integral coefficient is set to 0 for the hardware. Then using the previously determined MOI for the reaction wheel of 1.955 x 10^{-2} $\mathrm{kg \cdot m^2}$, and matching coefficients from the denominator of Equation 38, the PD gain settings were obtained where $K_p = 1.21153$ and $K_d = .109992$. Using these values, the resulting math model for the reaction wheel system is shown in Figure 24 below.

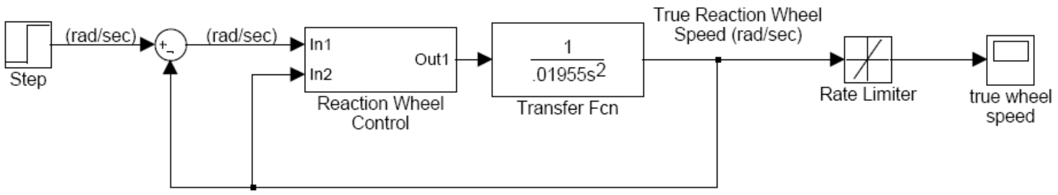


Figure 24. Reaction Wheel Math Model

The math model in Figure 24 was run at matching step commands and times in order to allow for a direct comparison between the math model and the hardware data. The results of this comparison are found in Section 4.4.

3.4.2 *Simplified Closed-Loop SimSat Model.*

The creation of a simplified closed-loop SimSat math model was necessary to ensure a truth source to compare against the data being obtained from the more complex models and the optimization technique. This equivalent second-order math model for SimSat was created using the following method.

- Create a closed-loop hardware model designed to be uploaded to SimSat that has a desired yaw orientation input in radians and then SimSat orients itself to the desired orientation. The resulting reaction wheel speed and orientation telemetry data are then captured in a Matlab® file. The actual model used in this effort is shown in Figure 25 below.

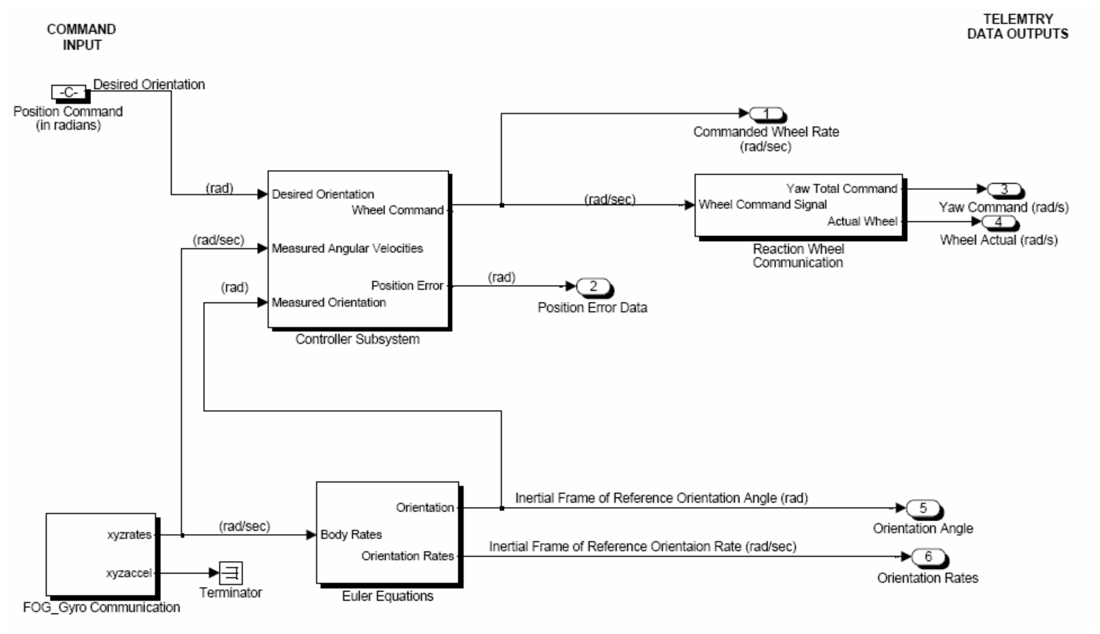


Figure 25. SimSat Hardware Model

- The model was run for a desired yaw orientation of 5, 10, 25, 45 and 90 degrees. This range of orientations captured SimSat's response to both small and large commanded orientation changes.

- The plots of the reaction wheel speeds and resulting orientation change were produced using the Matlab® code found in Appendix F. Plots for all five of the orientation changes can be found in Appendix G. The plot of SimSat's response to a 5° yaw commanded orientation change is found in Figure 26 below. This plot was used to obtain the basic transfer function of SimSat's response to a reaction wheel input.

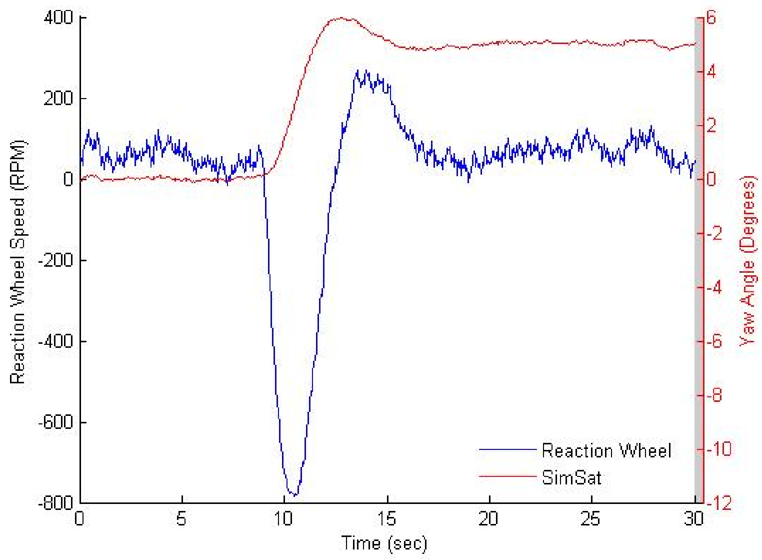


Figure 26. Plot of SimSat's Reaction Wheel Speed and Yaw Angle for 5° Orientation from Rest

- Through the use of Equations 27-30 and Figure 26 the second-order transfer function for SimSat could be obtained and is presented below

$$G_{SimSat}(s) = \frac{.632319}{s^2 + .761396s + .632319} \tag{39}$$

This transfer function was then placed in the math model shown in Figure 27 below; it was run for the same five orientation changes from the SimSat hardware. The results from this comparison are found in Section 4.5. It must be noted that this simplified equivalent second-order math model contains all of the information from the closed-loop hardware model of SimSat. This includes the reaction wheel controller, reaction wheel dynamics, and SimSat controller found in the closed-loop model in Appendix A. Accordingly, using a progressive approach (adding layers of complexity to each model), the development of the complex SimSat math models will be described in the following section.

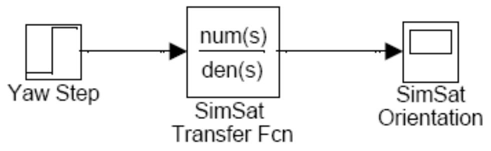


Figure 27. Simplified Closed-Loop Math Model of SimSat

3.4.3 ***Complex Closed-Loop SimSat Models.***

As mentioned in the previous section, a progressive approach was used to develop more accurate mathematical models of SimSat. The simplified closed-loop SimSat math model and the reaction wheel math model are essential groundwork in the development of the more complex closed-loop models. However, to determine a change in satellite propellant an accurate mathematical model of the SimSat hardware is necessary.

Starting from the simplified closed-loop model shown in Figure 27 and comparing it to the hardware model of SimSat (Figure 25), it is evident there is a PD controller used in the hardware. Therefore, a PD controller was added to the math model in order to remove the error between the desired and actual orientation angle. The addition of this PD controller and feedback changes the model as shown in Figure 28.

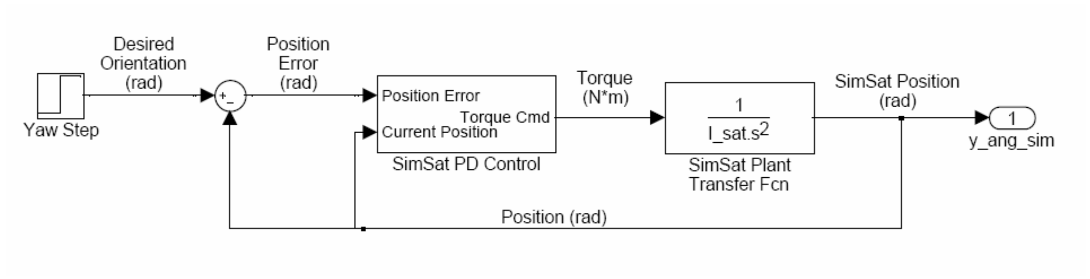


Figure 28. Math Model of SimSat Using PD Control

The PD controller in this model now represents the reaction wheel system and SimSat controller found in the hardware model. This refined model requires the knowledge of the MOI for SimSat in the yaw direction, determined in Section 3.3, in order to run. However, the MOI for SimSat can also be a variable. Therefore, the optimization techniques introduced in Section 2.6 can be used to determine the MOI from the acquired hardware data if needed. This approach, found in Section 3.5, is the basis for determining the change in satellite propellant using changes in MOI. To ensure the characteristics of the model were unchanged, this most recent model was compared to the previous model and SimSat hardware data previously obtained, these results can be found in Section 4.6.

Secondly, the reaction wheel system was added to the closed-loop SimSat model. This maneuver resulted in a significant challenge. The reaction wheel hardware for SimSat takes the current velocity and adds the commanded wheel rate change. In order to implement the hardware method in the math model and still obtain the required acceleration rate for the reaction wheel, the values must be created through a convoluted process. The true velocity of the reaction wheel in the math model is added to the previous value. This combined velocity is used to obtain the change in velocity for a time step interval in the model. The resulting acceleration is multiplied by the MOI of the reaction wheel to acquire the torque generated by the reaction wheel on SimSat. Torque is subsequently the input for the SimSat plant transfer function and the resulting output is the velocity of SimSat. By integrating the velocity for each time step, the position of SimSat can be determined. This model is shown below in Figure 29.

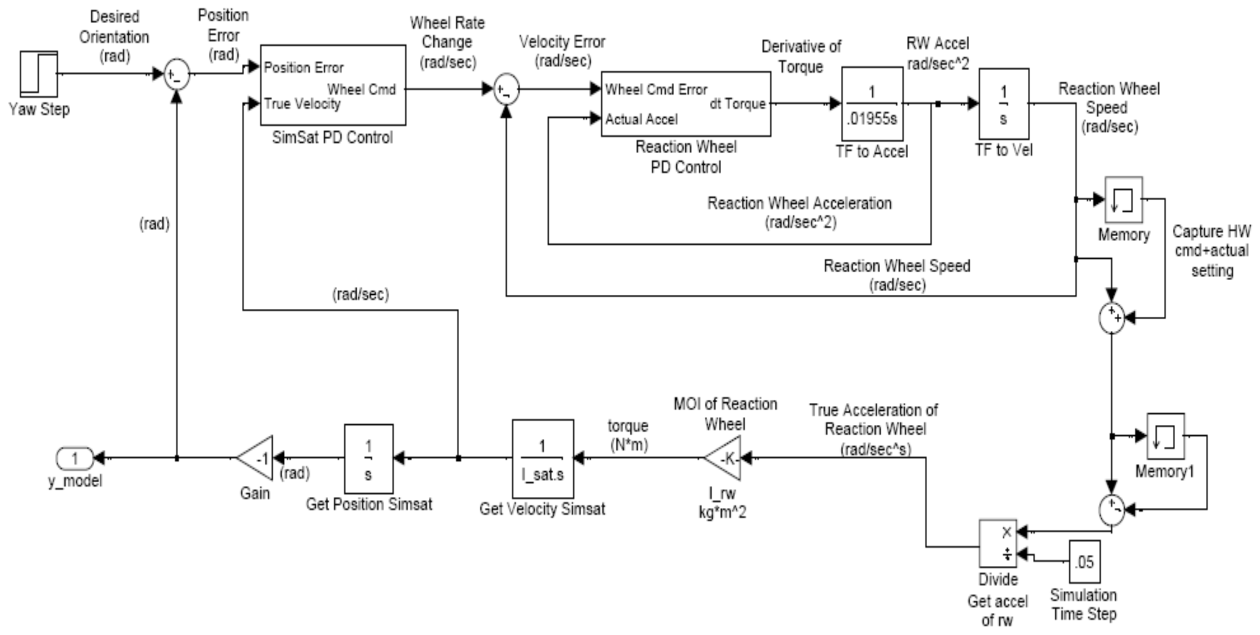


Figure 29. Math Model of SimSat Using PD Control and Reaction Wheel

The results from this model compared to the previous models and the actual SimSat data is found in Section 4.6.

The final step to develop a robust math model for SimSat is to use the coupled Euler’s Equations of Motion from Equations 26a-c to represent the satellite dynamics of

SimSat. This can be implemented in the closed-loop SimSat model since the reaction wheels are providing wheel acceleration, $\dot{\omega}_{rw}$, to the model. The satellite dynamic equations rely on reaction wheel acceleration, satellite velocity, and MOI to determine the satellite acceleration. Since these equations require three axis control, the reaction wheel system has been recreated for the pitch and roll axes. Additionally, the MOI in the pitch and roll axis are required for this model. The resulting math model is displayed in Figure 30. Section 4.6 presents the results of the orientation changes when the model was run against the SimSat hardware.

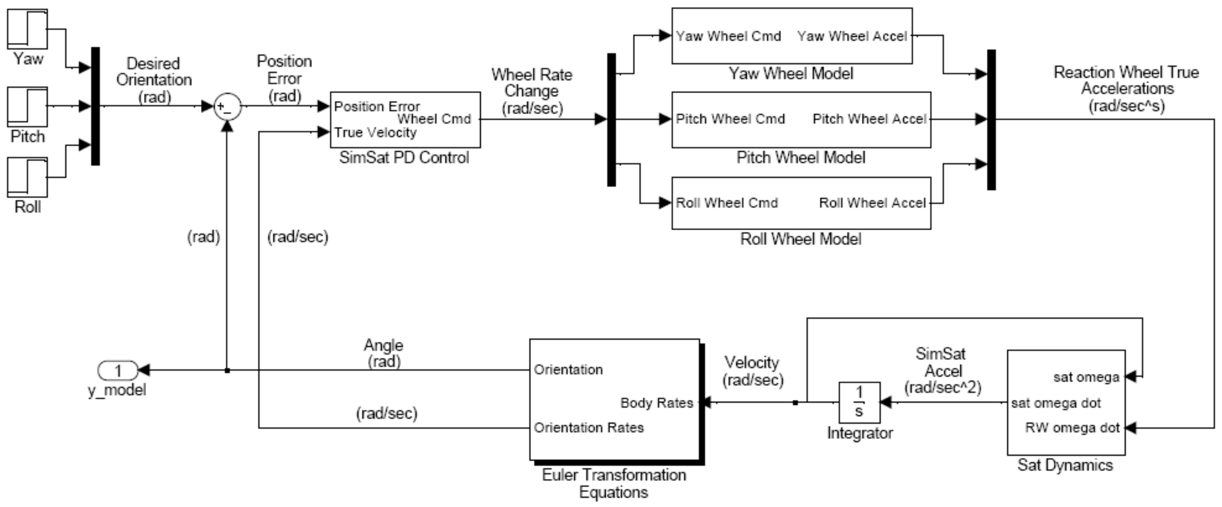


Figure 30. Math Model of SimSat Using Satellite Dynamic Equations

3.5 Determining the Change in Propellant from Measured MOI

The required components for detecting a change in satellite propellant from measured MOI were previously outlined. Next, the SimSat hardware configuration was changed and used to generate yaw orientation data for the new configuration. The analysis (by way of optimization techniques) determines the change in the MOI and thereby the change in propellant.

The first step in this process was to change SimSat's baseline MOI characteristics. Four .495 kg weights manufactured by AFIT's Lab were added to SimSat. Two of these weights were attached to each end plate on SimSat; adding a total of 1.5 $\mathrm{kg \cdot m^2}$ to the baseline MOI configuration. Then, using the SimSat hardware model from Figure 25, SimSat's orientation was changed from an at rest position to a 5° orientation change. The resulting yaw angle telemetry data was captured from the detection maneuver. Next, one weight from each end plate was removed to simulate the loss of fuel. The same detection maneuver was performed and the resulting data recorded. Finally, the process was repeated a third time with all of the weights removed to simulate a complete depletion of the fuel tanks. This final configuration is the baseline SimSat configuration.

In order to resolve what the MOI is for the experimental data, an optimization technique was employed following the theory presented in Section 2.7. A Matlab® code was written to import and then minimize the cost function between the experimental and model data. The cost function used in this method is found below

$$min_p \ \mathrm{J}(p) = \sum_{i=1}^{n} \left| \overline{\theta}_i - \theta_i\left(p\right) \right|^2 \tag{40}$$

where the experimental data is $\overline{\theta}_i$, the model data is θ_i, and the design parameter *p* is the unknown MOI which is being determined over *n* set of data points. The *Find_Hardware_MOI* Matlab® code in Appendix H accomplishes the minimization of

Equation 40 by first loading the desired experimental data to use in the cost function. Next, one of the closed-loop SimSat math models found in Figure 28, 29, or 30 is called and run using the initial guess for the MOI value. The resulting model data is used in Equation 40 to determine a cost value. Then, the MOI value is iterated upon and the math model is rerun until the cost function converges to a solution. The resulting MOI found using the optimization technique provides a best fit to the experimental data's MOI using the math models given constraints. The results of this methodology and its applicability in determining the change in satellite propellant will be discussed in Section 4.7.

3.6 Summary

This chapter described in detail the experimental equipment used as part of this research. An improved reaction wheel math model along with several closed-loop SimSat models were presented in order to frame the process for determining the change in MOI. Finally, the method used to determine MOI and remaining satellite propellant was discussed. Chapter IV will include the results of this methodology as well as its limitations for applicability. Finally, Chapter V will outline the conclusions from this research and highlight directions for future research.

IV. Results and Analysis

4.1 Chapter Overview

This chapter describes and analyzes the results obtained from the theory and applied methodology described in Chapters II and III. Overall results include the determination of the fiber optic gyroscope drift rate, baseline MOI for SimSat, improved reaction wheel model comparison, closed-loop SimSat model comparisons, and the capability to determine the change in Satellite propellant using MOI data. Several attempts to improve the closed-loop model response to obtained hardware data were also accomplished. Each separate category of the research is described in detail in the following text.

4.2 Fiber Gyroscope Drift Rate

The first step to ensure the accuracy of the fiber optic gyroscope data collection was to give SimSat a commanded orientation change and analyze the resulting telemetry data provided by the gyroscope. This was accomplished as described in Section 3.2.2.2. To recap the process; SimSat was commanded to an orientation of +90° from an at rest position, and then held at the desired orientation until SimSat suffered a gyro or wheel malfunction. The initial and final positions, illuminated by the laser pointer, were marked on a piece of paper on the wall. This process was repeated 7 times to ensure accuracy of the data being collected and to provide some confidence in the values being obtained. The resulting yaw telemetry data and the position error (difference between the actual and desired yaw angle) were plotted versus time as seen in Figure 31 below.

Using the Matlab® script in Appendix I, plots for all 7 data runs were created. The plots visibly show that the data collection is terminated once the gyro or wheel malfunctions.

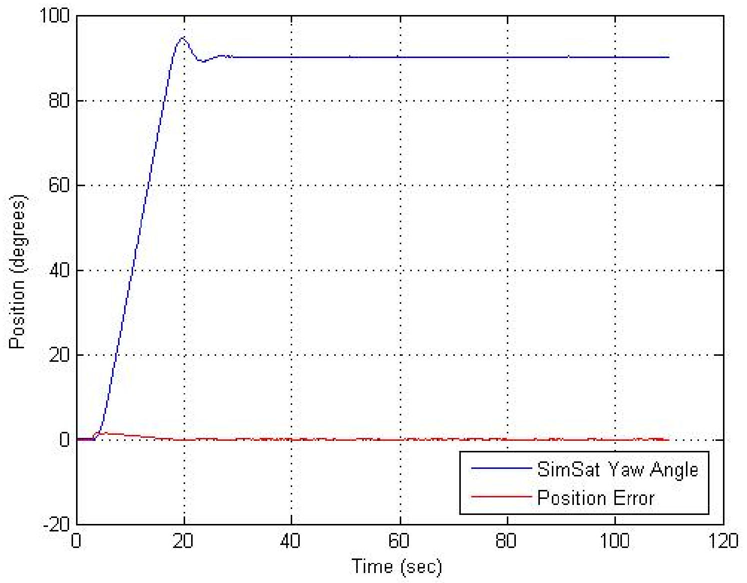


Figure 31. SimSat Yaw Angle and Position Error for +90° Orientation Change

While the gyro appeared to hold the commanded position, in actuality SimSat drifted during the hold maneuver. The drift rate of the gyro, and subsequently SimSat was determined by calculating the initial and final position of SimSat through geometric means from the laser designated data collected during the experiment. Using the difference between the two true orientations and the amount of time SimSat held the drifting position, the resulting degree of drift per second were calculated. The results from this analysis are shown in Table 4 below.

Table 4. Calculated Drift Rate of Fiber Optic Gyroscope

Run	True Starting Position (Deg)	True End Position (Deg)	Degree of Drift From Start to Finish	Time for Drift (sec)	Degree Per Sec of Drift
1	90.7533	88.2621	2.4911	80	0.0311
2	90.2318	87.7991	2.4327	44.6	0.0545
3	90.5794	85.8928	4.6866	156.6	0.0299
4	90.2318	88.9571	1.2747	31.7	0.0402
5	90.5215	88.7254	1.7961	78.6	0.0229
6	90.1159	88.0306	2.0853	149.1	0.0140
7	90.9850	89.8841	1.1009	169.4	0.0065

From the results shown above, the average drift rate for the gyroscope is 0.0285 degrees per second. This equates to an approximate drift rate of 1.7^{o} per minute, which is significantly higher than the previously quoted values (Table 3) for the LN-200 Fiber Gyro. This drift rate is an "effective drift rate" for the SimSat hardware and is a result of the digital sampling and integration of the values, which due to noise specifically bias the results from the experiment to be much higher than the device only drift rate. A substantial SimSat drift rate is only noticeable when SimSat is commanded to hold a specific position for a considerable period of time (> 300 sec). However, the latter is not an issue for this project as the author is only concerned with SimSat's ability to transition from an at rest position to a desired orientation. Therefore, the drift rate calculated in this section should have minor impact on this research effort.

4.3 Baseline MOI for SimSat

As mentioned previously in Section 3.3 the baseline MOI of SimSat are required for use in calculating the initial gain values for the PD controller used in SimSat's closed loop math model. Additionally, the baseline MOI values for SimSat can be used to verify the accuracy of the math models created as part of this research. Using the method outlined in Section 3.3, along with the *Determine_MOI_SimSat_3_Axis* Matlab® code in Appendix B, the MOI for SimSat were determined about all three axes. The code was run for the open loop data sets containing valid gyro and reaction wheel telemetry data. The resulting MOI in each of the three axes was recorded and the average of each axes MOI was calculated. Utilizing the average of the MOI values helps eliminate any bias that may have occurred in the data collection methodology. SimSat's MOI matrix, in the form of Equation 22, was found to be

$$\mathbf{I}_{SimSat} = \begin{pmatrix} 4.898 & 0 & 0 \\ 0 & 37.211 & 0 \\ 0 & 0 & 45.885 \end{pmatrix} \mathrm{kg \cdot m^2}$$

which means the MOI about the yaw axis is 45.885 $\mathrm{kg \cdot m^2}$, about the pitch axis is 37.211 $\mathrm{kg \cdot m^2}$, and about the roll axis is 4.898 $\mathrm{kg \cdot m^2}$. These baseline MOI for the three axes will not vary unless weight is added or removed from the system. Since these values are considered constant they provide a good truth source to verify the accuracy of the math models created in this research. Additionally, these values are consistent with Dabrowski's previously obtained MOI values using a different method. The roll and yaw MOI found in this research are 30% higher while the pitch MOI is only 2% higher from the previously calculated values. This is due to several reasons, the addition of hardware to SimSat will impact the MOI calculations. As mentioned previously the addition of a fiber optic gyro and camera system have been accomplished since Dabrowski's work. Also, the method used to determine the MOI were different. Dabrowski used an average acceleration of SimSat and the reaction wheel to determine the MOI in the three axes which is different than the method employed in this research. Using the baseline MOI, the gain values for SimSat's closed loop PD controller can also be obtained. However, since this research is concentrated in the yaw direction of motion, the values for the pitch and roll portion of the PD controller were not calculated. The gain values calculated for the yaw portion of SimSat's PD controller can be found in Section 4.6 below.

4.4 Reaction Wheel Response

In order to characterize the response of SimSat's reaction wheel to a commanded input, the methodology presented in Section 3.4.1 was used on the yaw reaction wheel. Plots of the resulting response of the reaction wheel to a commanded input are provided in Appendix E. The author initially thought that the reaction wheels would behave linearly when responding to a command. However, from Figures 32 - 34 below it becomes quite clear the reaction wheels do not respond linearly, they appear to have a rate limiter, impacting the swiftness of wheel response to a commanded input. In Figure 32 there is no rate limiter effects noted for the 5 rad/sec input due to the large overshoot of the wheel and the .07 sec rise time it takes for the actual wheel to respond to the command. Whereas, in Figure 33 the rate limiter has significantly reduced the overshoot; however, the rise time has increased to .55 sec instead of the anticipated .35 seconds for a linear system. The difference in the rise time has a minimal impact on the system since the settling time for both of the inputs is ~3 sec. As the input was increased, it was noted that the rate limiter appeared to have a maximum setting. This can be seen in the 50 – 80 rad/sec plots found in Appendix E. The max rate limit setting seems to condition the response of the reaction wheel to mirror the commanded step input, little to no overshoot, and only minor changes in the response time of the wheel depending on the input. This smoothed response of the reaction wheel at the max rate limit is shown in Figure 34 below.

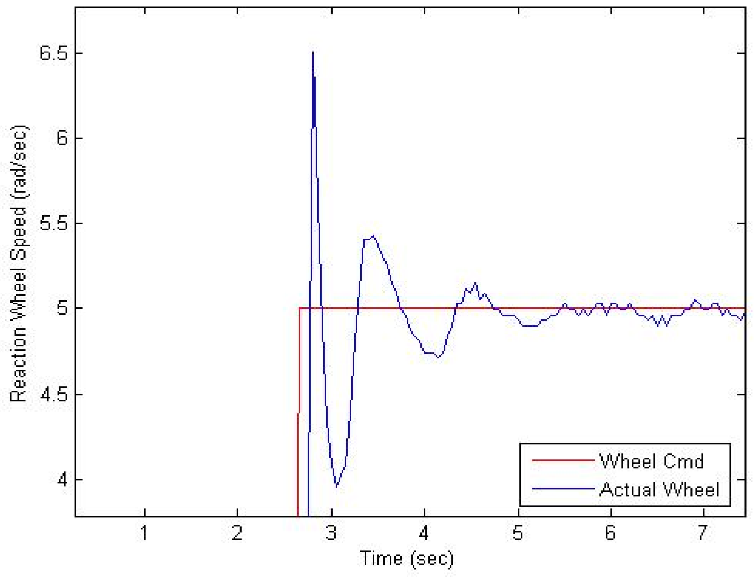


Figure 32. Reaction Wheel Response to a 5 rad/sec Step Input

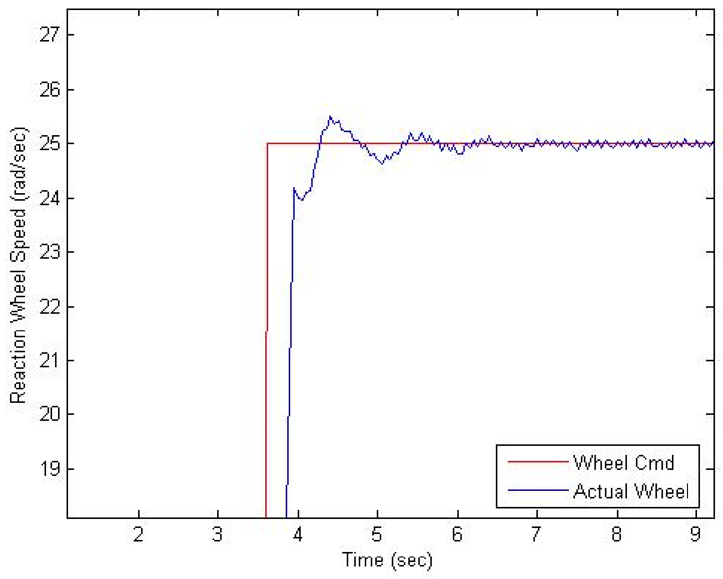


Figure 33. Reaction Wheel Response to a 25 rad/sec Step Input

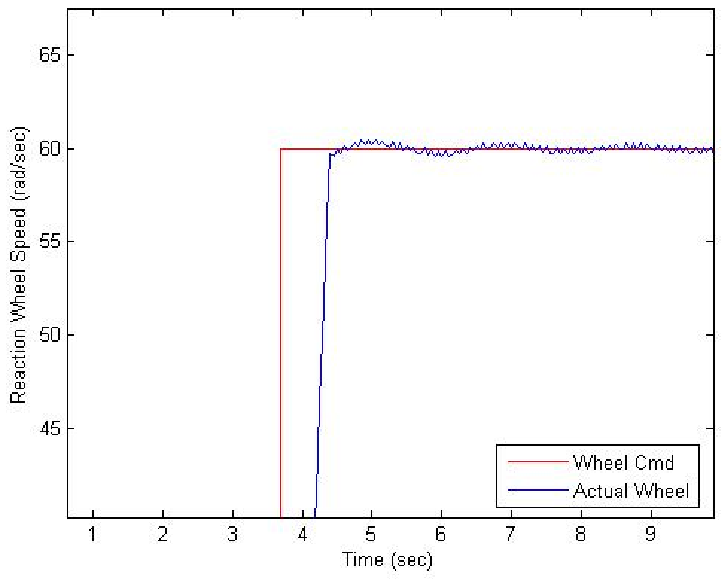


Figure 34. Reaction Wheel Response to a 60 rad/sec Step Input

Due to the effects of the rate limiter, the 5 rad/sec plot shown in Figure 32 was used to calculate the transfer function of the reaction wheel (reference Equation 38).

Using the 5 rad/sec reaction wheel setting ensured little to no rate limiter effects would be included in the transfer function, allowing for the creation of a more accurate math model of the reaction wheel. The rate limiter effects could be added to the model once a clear baseline was established.

Using Equation 38 and the PD theory, the math model for the reaction wheel was created and previously shown in Figure 24. Next, the model was verified for accuracy against the hardware data that was previously obtained. The model was run at the same input and time settings as the hardware data to allow for a direct comparison. The 5 rad/sec setting, shown below in Figure 35, shows a close comparison between the actual and model system. The overshoot and rise time are similar while the settling time for the model is approximately 1.5 seconds faster than the actual system.

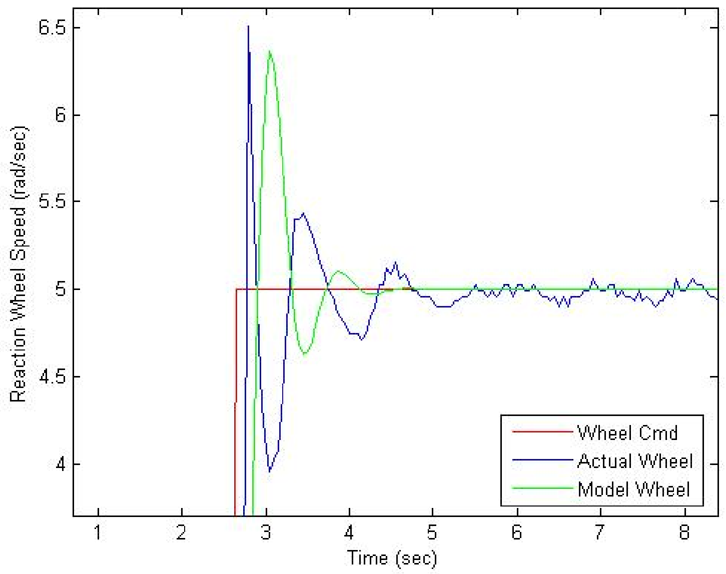


Figure 35. Comparison of Reaction Wheel Model and Actual Data for 5 rad/sec Input

However, as the input was changed in 5 rad/sec increments, the model required the rate limiter setting to be changed in order to make the modeled data more representative of the actual data. This time intensive process was required for the 10 – 50 rad/sec settings due to the nonlinear nature of the reaction wheel. The resulting comparison between the model and actual data became more exact for the increasing reaction wheel input. This is shown in Figures 36 and 37.

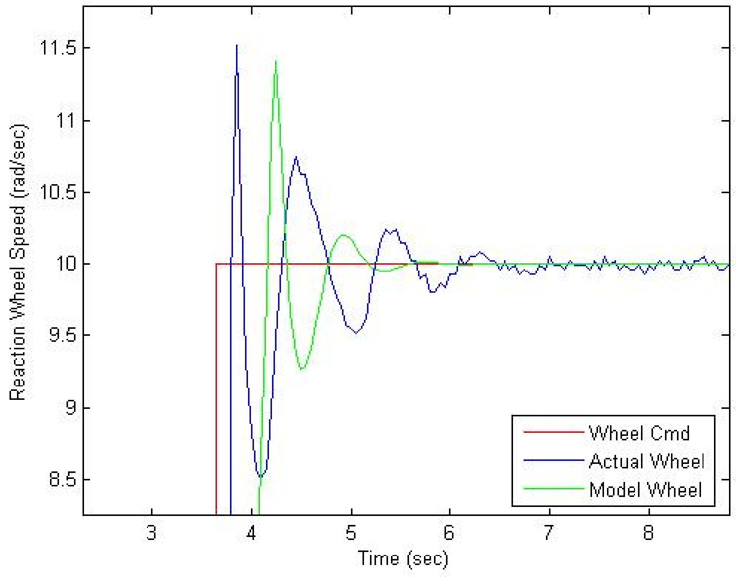


Figure 36. Comparison of Reaction Wheel Model and Actual Data for 10 rad/sec Input

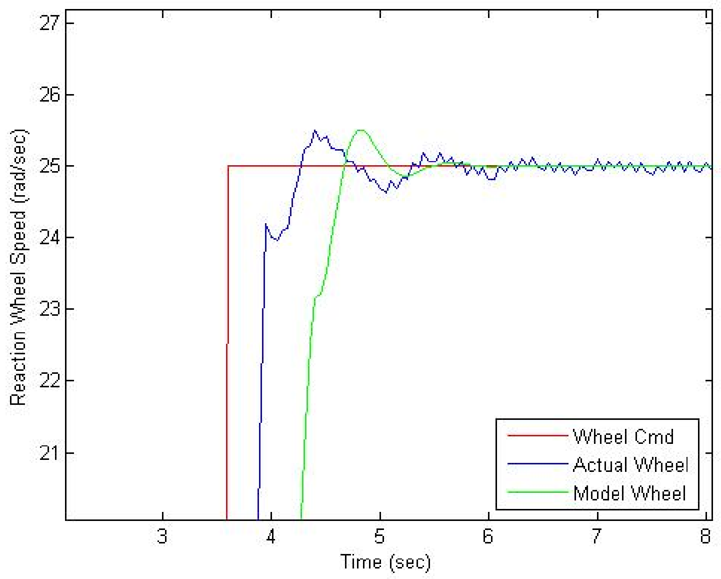


Figure 37. Comparison of Reaction Wheel Model and Actual Data for 25 rad/sec Input

Once the reaction wheel input went above 50 rad/sec, changes to the rate limiter setting were no longer required, the modeled data compared very well to the actual data. The resulting comparison for the 55 – 80 rad/sec is similar to the one seen in Figure 38. The primary difference between the modeled and actual data is the rise time. The model takes around 0.5 sec longer to reach the desired reaction wheel speed compared to the actual hardware, but this additional time is not significant in relation to the overall matching of the two systems. Overall the reaction wheel math model is an accurate representation of the hardware if it is tuned properly for the anticipated wheel speed being used.

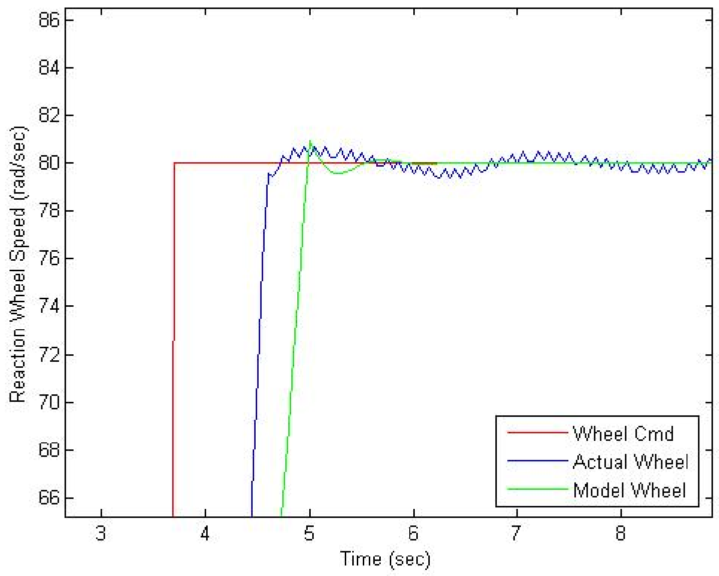


Figure 38. Comparison of Reaction Wheel Model and Actual Data for 80 rad/sec Input

4.5 Simplified Closed-Loop SimSat Math Model

In order to ensure a good truth source to use as a baseline for SimSat's response to an input, an equivalent second-order math model for SimSat (see Figure 27) using the transfer function in Equation 39 was created. This simplified closed-loop SimSat model should provide a relatively good representation of SimSat's response to a commanded orientation change. At a minimum it should ensure a similar type response to the values obtained from the hardware and provided orientation angle plots similar to those found in Appendix G.

The simplified closed-loop math model was run for the corresponding hardware orientation changes of 5º, 10º, 25º, 45º, and 90º and compared to the actual hardware values. The math model for the 5º and 10º change, shown below in Figures 39 and 40, corresponds accurately with the hardware data. The 5º case is the most accurate due to the transfer function for the math model being developed from the corresponding hardware data. In the 10º comparison, the model and hardware data differ slightly in the max overshoot orientation accompanied by a slight variance in the time to settle on a final orientation.

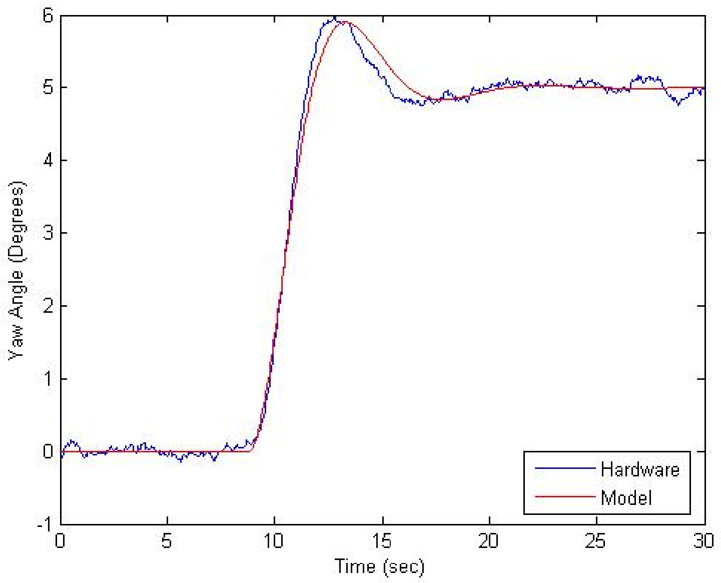


Figure 39. Simplified Closed-Loop SimSat Math Model vs Hardware Data for 5º Orientation Change

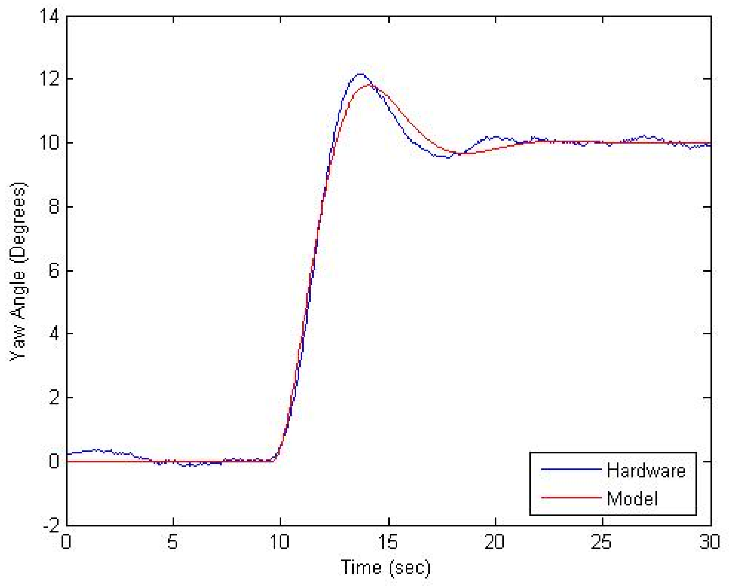


Figure 40. Simplified Closed-Loop SimSat Math Model vs Hardware for 10º Orientation Change

While the 5º and 10º math models provide an accurate comparison between the model and hardware data; the 25º, 45º, and 90º math models are not useable as a baseline model for SimSat. These comparisons shown in Figures 41 - 43 show the models differ significantly from the actual hardware data. The primary reason for this difference is these models have consolidated the reaction wheel controller, reaction wheel dynamics, SimSat controller, and other constraints that are required in the math model to make a more realistic match between the math model and hardware system. Therefore, using the progressive approach as outlined in Section 3.4 for the development of the closed loop SimSat math model should produce a model capable of simulating SimSat's hardware motion as it is given any commanded orientation change.

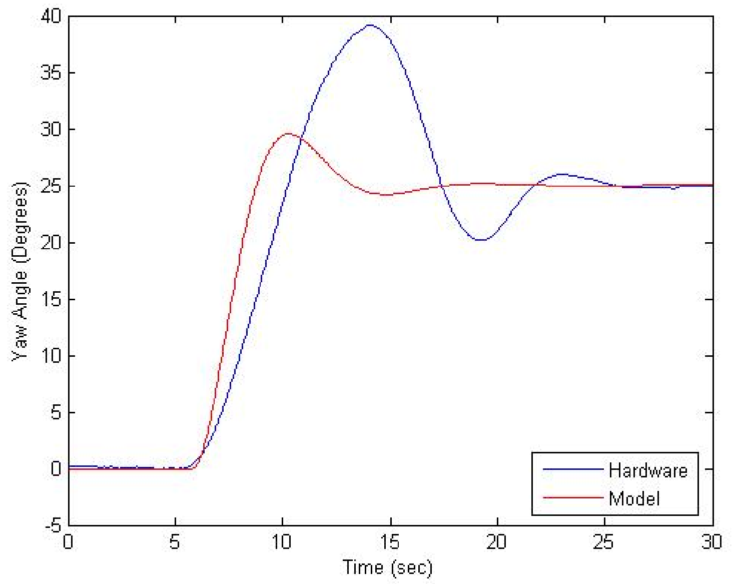


Figure 41. Simplified Closed-Loop SimSat Math Model vs Hardware for 25º Orientation Change

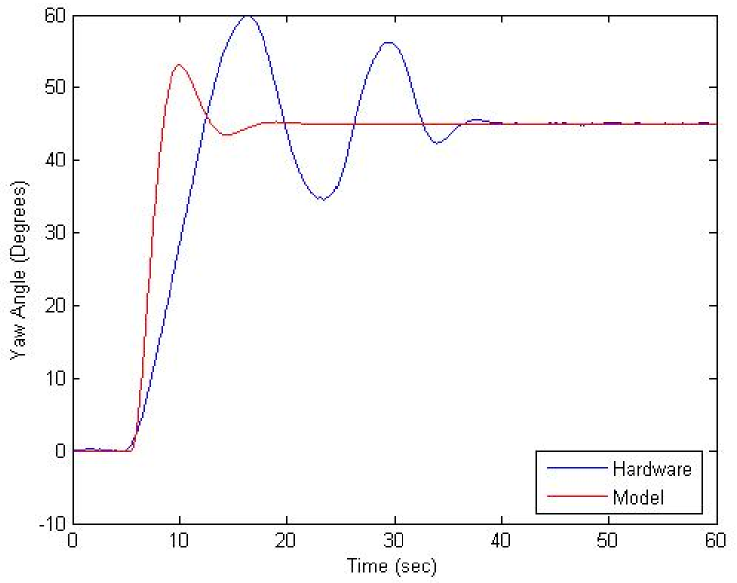


Figure 42. Simplified Closed-Loop SimSat Math Model vs Hardware for 45º Orientation Change

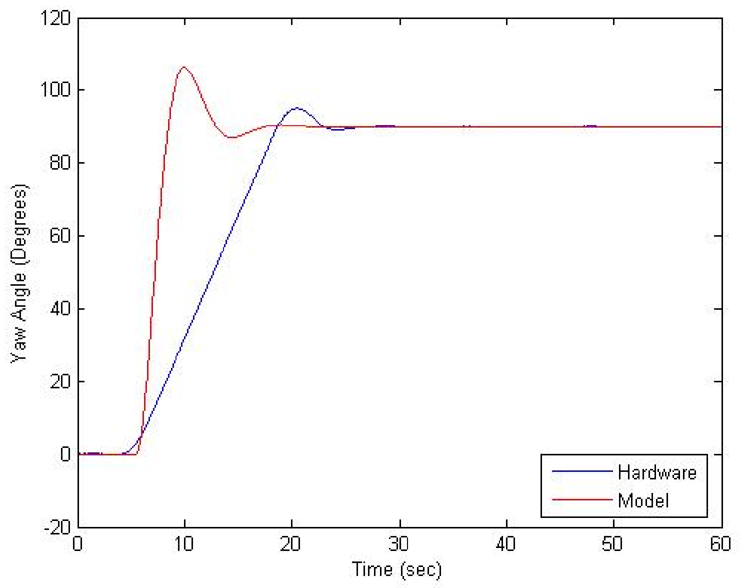


Figure 43. Simplified Closed-Loop SimSat Math Model vs Hardware for 90º Orientation Change

4.6 Complex Closed-Loop SimSat Math Models

Moving from the simplified closed-loop model of SimSat to a more complex closed-loop model permits a more accurate depiction of SimSat's different systems. These additional layers of fidelity to the math models allow more parameters to be tuned to present a more accurate representation of SimSat and how it responds to a desired orientation change. The addition of a PD controller to the closed-loop SimSat model allows for the determination of the MOI of SimSat through mathematical optimization. Using SimSat's yaw MOI, found in Section 4.3, the baseline PD controller gains could be calculated. These gains were found to be $K_p = 29.0142$ and $K_d = 34.9369$ and were inserted into the model from Figure 28. The PD/SimSat model was then run for a 5º, 10º, and 25º desired orientation. When the results from the PD/SimSat model were compared against the previous simplified closed-loop SimSat model results for the same orientation changes the two sets of orientation data matched. This proved that the addition of the PD controller to the math model did not impact the overall model characteristics and the method used to add the PD controller to the model was valid. However, even though the results from the PD/SimSat closed-loop model matched the initial simplified closed-loop SimSat model, the baseline settings for K_p and K_d were not accurate enough to determine the baseline yaw MOI through optimization. This required the math model PD controller gains to be changed. The most efficient way to match the model and actual data was through the use of optimization techniques. Using the theory outlined in Section 2.6 the PD gains K_p and K_d were used as variables that could be iterated upon in order to minimize the difference between the obtained model data and the hardware yaw data, while the MOI remained a fixed value in the model. The minimization was accomplished using the *Find_PD_Gains* Matlab® code in Appendix J. The code requires the reference data file to be selected and using the model from Figure 28 determines the best K_p and K_d that would minimize the cost function using, and therefore match the fixed MOI provided in the model. The optimization technique uses the entire 30 second time data to determine the optimized values. This optimization process was carried out on the 5º orientation change data and resulted in the gain values of $K_p = 34.0399$ and $K_d = 36.599$. These 'equivalent' controller gains adjust for other errors not accounted for in the model. Using these 'equivalent' gains produces an extremely accurate math model and when compared to the hardware data, as seen in Figure 44, allowed for the calculation, using optimization, of the previously obtained yaw MOI. This proves that if the math model is accurate the MOI of SimSat can be determined given some orientation change telemetry data.

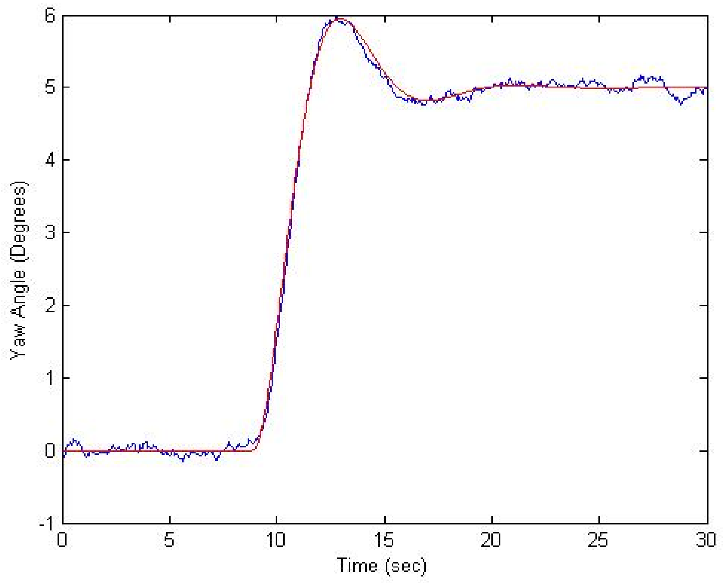


Figure 44. Comparison of Hardware vs Optimized Math Model for 5º Orientation Change

Using the newly optimized math model the 10° case was rerun and compared to the hardware data. The results from the comparison can be seen in Figure 45.

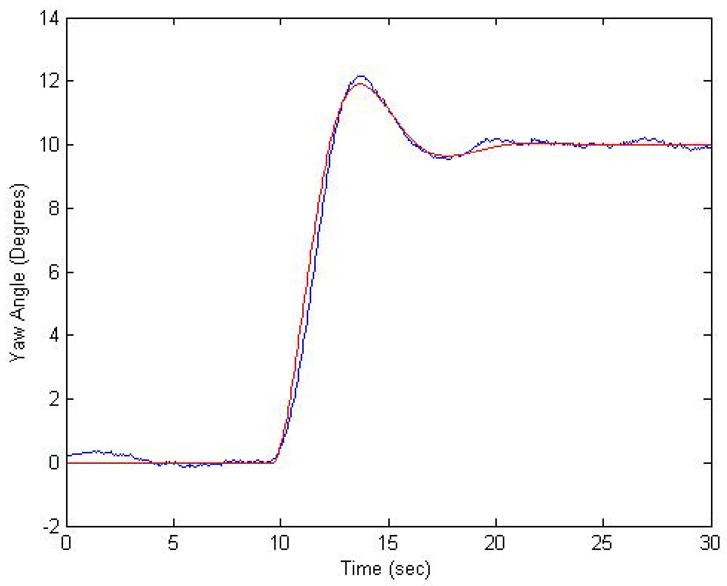


Figure 45. Comparison of Hardware vs Math Model for 10º Orientation Change

While this new model improved over the 10° simplified closed-loop SimSat transfer function model, as seen in Figure 40, the modeled yaw orientation change data did not allow for an accurate determination of the baseline MOI. The MOI the minimization routine calculated was 10% lower than the baseline MOI. This meant that the 10° model would also have to have an 'equivalent' set of controller gains calculated using the same 30 second time interval of data to provide a more accurate MOI. The comparison between the hardware and optimized 10° model using the 'equivalent' controller gains is shown in Figure 46.

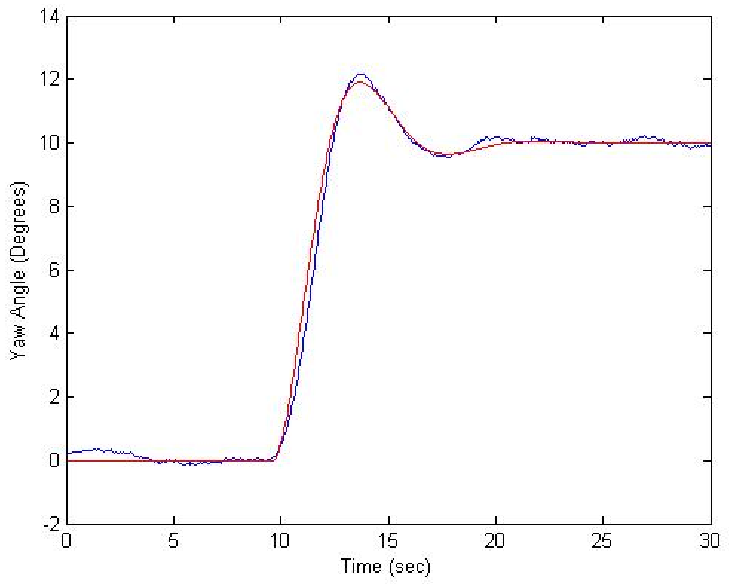


Figure 46. Comparison of Hardware vs Optimized Math Model for 10º Orientation Change

This optimized model was also run for the 25°, 45°, and 90° orientation change and when compared against the hardware data were extremely inaccurate as seen in Figure 47. The optimization process previously used on the 5° and 10° runs was attempted on these orientation changes, but was unsuccessful. Therefore, the math model currently in use is not adequate to handle all of the orientation changes applied to SimSat and the next iteration of the math model was created.

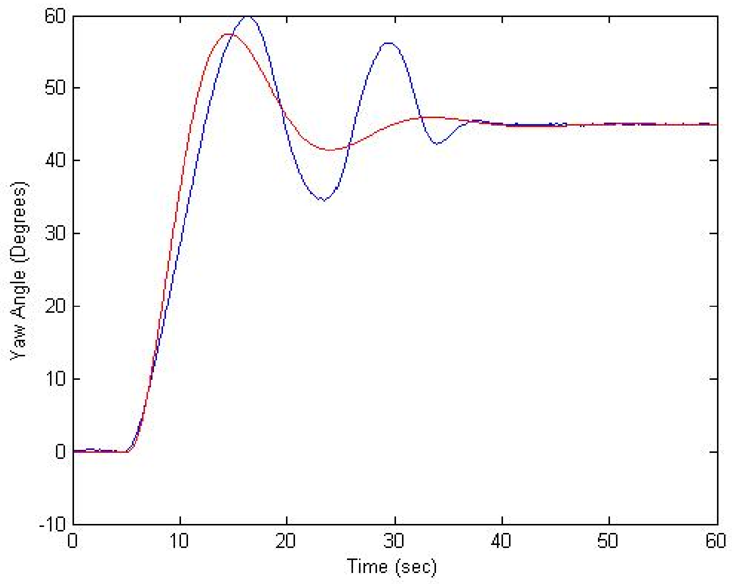


Figure 47. Comparison of Hardware vs Math Model for 45° Orientation Change

This next progression in the creation of a robust math model of SimSat combined the previously created PD/SimSat closed-loop model and the reaction wheel model. This combined PD and reaction wheel (PD/RW) model, shown in Figure 29, captures more of SimSat's characteristics and provides more parameters that can be adjusted to match the math model with the resulting hardware data. Using the 'equivalent' gain values from the 5° math model and the reaction wheel gain values the PD/RW model was run for a 5° orientation change. The results for this baseline run are shown in Figure 48 below.

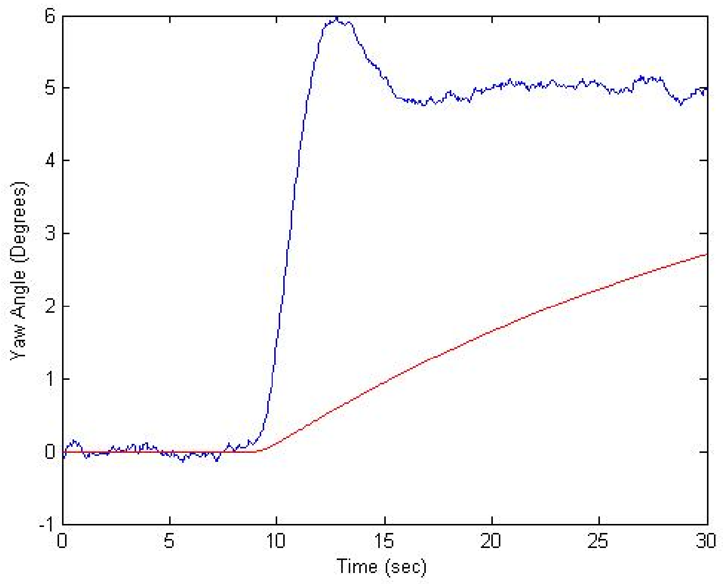


Figure 48. Comparison of Hardware vs PD/RW Model for 5º Orientation Change

The gain values previously calculated for the reaction wheel PD controller and SimSat PD controller when combined produce an extremely inaccurate model. This is due to the nonlinearity of the reaction wheel that is not adequately captured in the previous model. Therefore, the model will have to be tuned using optimization techniques previously discussed in this thesis for each desired orientation change angle. Several different methods were used to try and optimize the response to the different orientation change baseline data that has been collected. The first method attempted to use a completely unconstrained problem with four variables, the gain values found in the PD controllers. This method was unsuccessful and after several weeks of attempting to match the model to the hardware data a more constrained optimization program was attempted. The gain values for the reaction wheel PD controller were fixed at their baseline settings and only the SimSat PD controller gains were allowed to vary. This resulted in the PD/RW model providing an exact match to the 5° yaw hardware data, and is shown in Figure 49 below.

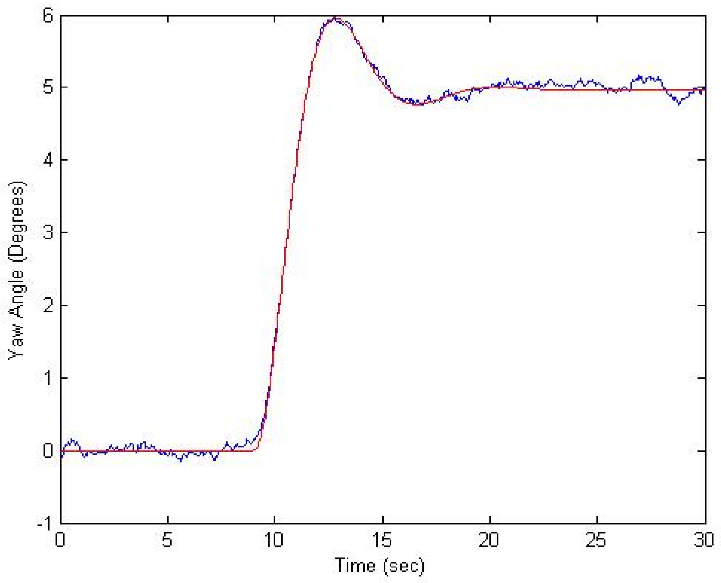


Figure 49. Comparison of Hardware vs Optimized PD/RW Model for 5º Orientation Change

The only noticeable difference in the figure between the hardware and model data is what appears to be noise in the hardware orientation signal. This is the result of SimSat trying to maintain the initial at rest orientation by changing the reaction wheel velocity and as a result SimSat's orientation varies slightly. The same orientation stabilization issue occurs as SimSat maintains the commanded 5° orientation change and adjusts the reaction wheel speeds to maintain the orientation. This is clearly seen in Figure 50 below where the model reaction wheel speed is compared to the data from SimSat. The model provides a reasonable approximation of the real reaction wheel speed. Also, the figure shows how the reaction wheel is constantly changing speed and direction in order to maintain the desired orientation. This constant motion of the reaction wheel results in the extraneous motion identified previously in Figure 49.

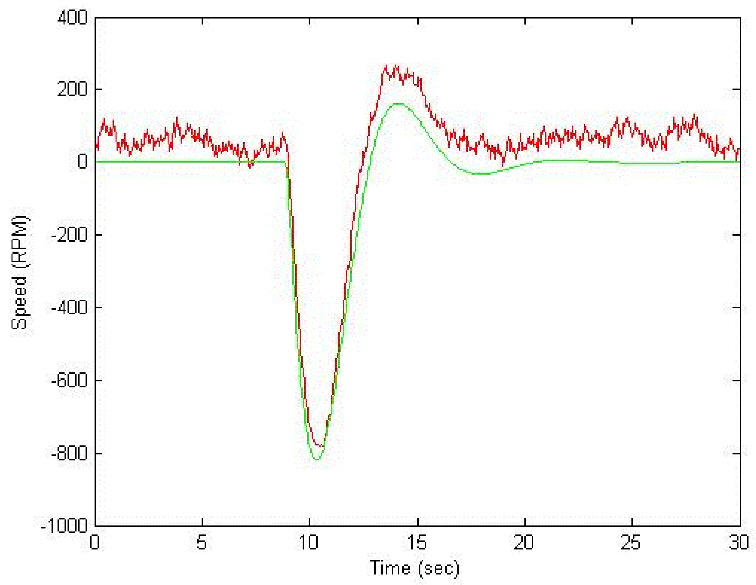


Figure 50. Comparison of Hardware vs Model Reaction Wheel Speed for 5º Orientation Change

The 10° orientation change was accomplished in the same manner and is shown in Appendix K. The model data for both the yaw orientation change and the reaction wheel response are just as accurate as the 5° results. However, the 25°, 45°, and 90° orientation change maneuver could not be replicated by the model. The math model for the large angle maneuvers was optimized using four variables allowing all four gain values to iterate. When the four variable iteration process failed the reaction wheel PD gain values were again fixed at their known values. The optimization process was repeated using just two variables and was unsuccessful in matching the model and hardware response as was done in the 5° and 10° runs. Figure 51 below shows that the PD/RW model will be insufficient for determining the MOI change using a 25° orientation change due to the large discrepancy between the model and hardware data. The 45° and 90° optimization results in Appendix K show the same discrepancy between the model and hardware data. The 90° results show the effects of an additional rate limiter that was not previously identified in the hardware model. This can be seen in the reaction wheel speed being held constant at around 2200 RPM for over 10 seconds while SimSat continued to change orientation. The addition of a rate limiter may correct some of the discontinuities between the model and hardware response for the large orientation maneuvers.

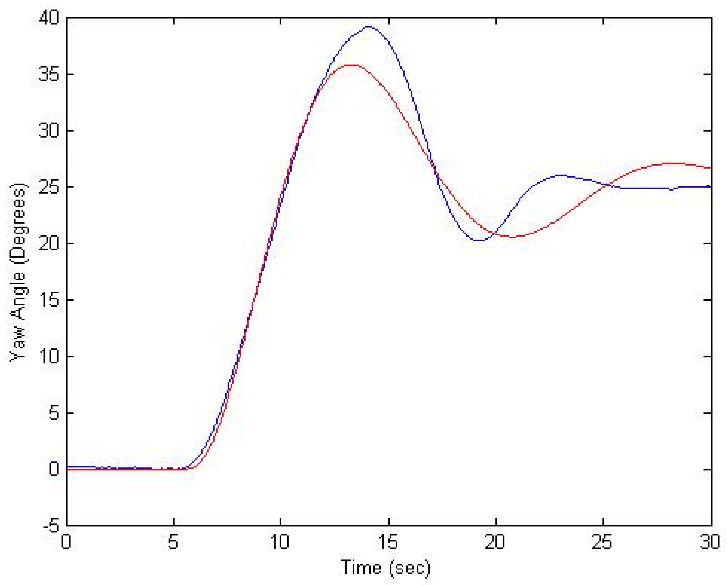


Figure 51. Comparison of Hardware vs Optimized PD/RW Model for 25^o Orientation Change

The final closed-loop model to discuss is the full satellite dynamic model. This math model using the coupled satellite dynamic equations of motion (Figure 30) was not able to be completed in this research effort and is left for future efforts. Therefore, the

PD/RW model will be used to determine if the change in satellite propellant can be determined from MOI data obtained from SimSat.

4.7 Determination of Satellite Propellant Using MOI

Since the previous section showed the closed-loop SimSat models using a 25° or greater change in orientation were inaccurate when compared to the hardware data, this research chose to use a small (5°) orientation change for the detection maneuver. Using the methodology outlined in Section 3.5, the baseline MOI characteristics of SimSat were changed through the addition of two weights to each end of SimSat. SimSat was then commanded from an at rest position to the desired orientation. Next, one weight from each end of SimSat was removed and it was commanded to the same desired orientation. Finally, the clean configuration of SimSat with no weights was commanded to the desired orientation. SimSat did not require rebalancing as the weights were removed due to the initial placement of the weights taking into account there eventual removal and impact to SimSat's balance. Each configuration had three data collection runs. The new baseline configuration of SimSat, with 2 kg of weights added, also had the baseline MOI determined using the open-loop method previously discussed in this thesis. This permitted the PD/RW model to be tuned to match the new baseline MOI data of SimSat using the 30 seconds of data collected. Once this model was tuned all of the design parameters were held constant, and only the MOI for the system was allowed to be iterated upon during the optimization routine. This resulted in the MOI for SimSat with 1 kg of weight added and of the clean configuration to be ascertained. The resulting data from the optimization routine and the baseline MOI is shown in Table 5 below. A plot showing the yaw orientation data for the Run #1 data set is shown in Figure 52 below.

Table 5. Calculated MOI for Determining Change in Satellite Fuel

	Baseline MOI for SimSat 2 kg of Weight	**MOI with 1 kg of Weight**	**MOI of Clean Configuration**
Run #1	48.194	46.848	46.267
Run #2	48.780	46.632	45.269
Run #3	48.235	46.008	45.772
Average MOI Value	**48.403**	**46.496**	**45.769**

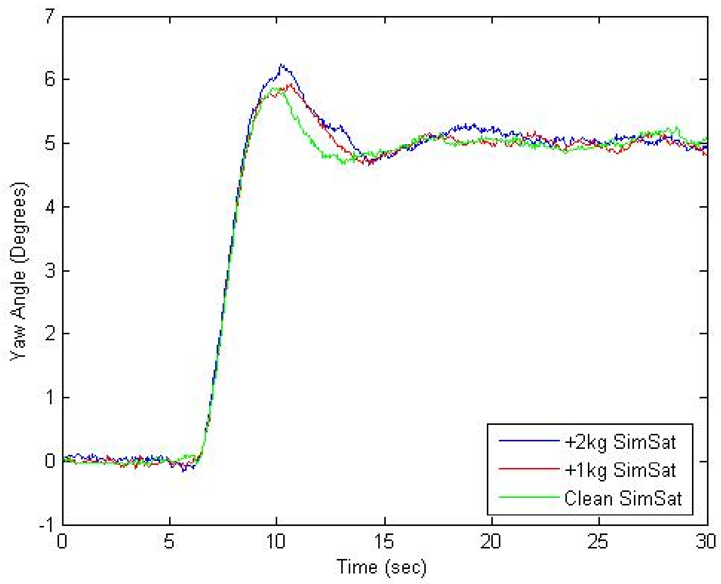


Figure 52. Plot of Run #1 Yaw Orientation Data for 3 SimSat Configurations

The data obtained from the baseline MOI calculations is relatively consistent; all of the values are within ±2% of each other. Thus, it can be concluded the baseline calculations were accurate in determining the MOI from the open-loop SimSat data. The MOI determined with the 1 kg total weight is also within ±2% of each value. The clean configuration is within ±2.2% . These results show that the data within each MOI configuration are consistent with each other, proving the optimization technique utilized in this research provided consistent values for the unknown MOI. However, looking at the limited sample size, the only definitive conclusion that can be drawn is that the difference between the baseline and clean configuration provides a method to determine the change in fuel for a 5° maneuver. While the change in MOI between the baseline and clean configuration is $2.634\,\mathrm{kg}\cdot\mathrm{m}^2$, this value is 150% greater than the calculated difference in MOI. The addition of 2 kg of weight and the weights placement on SimSat resulted in the addition of 1.5 $\mathrm{kg}\cdot\mathrm{m}^2$ to the MOI. Therefore, the optimized results between the baseline SimSat configuration and clean configuration yielded a larger MOI difference than was physically placed on SimSat. This extra MOI calculated in the optimization technique could have occurred due to the gyro drift effects previously mentioned in this thesis. Using small orientation movements and holding those positions for almost 30 seconds as part of this research would add ~.85° to the yaw angle SimSat was commanded to hold. This additional angle will impact the optimization results because this research used the entire 30 seconds of collected orientation data to optimize over and determine the MOI for the data set. The MOI for the 1 kg weight setting compared with the clean configuration are within 2% of each other. Therefore, it can not be concluded if the difference between the two test setups is due to the added mass or experimental error. While the first and second runs show a substantial difference between the two configurations, the third data run has the two configurations almost at the identical MOI. This could be due to several different factors, but when averaged using the other values no determination can be concluded. Overall, the method used to detect the change in weight on SimSat using MOI is valid. However, it appears from the limited data the method using the currently tuned models can only detect a change greater than 2 kg total mass.

V. Conclusions and Recommendations

5.1 Chapter Overview

Without additional methods to determine the remaining fuel onboard an orbiting satellite, companies are losing millions of dollars for the premature retirement of the satellite. Additionally, the military has limited space assets available and must be diligent in prolonging the life of a satellite in operation. The ability to determine the remaining propellant through measure MOI is a step in providing this increased detection ability.

5.2 Conclusions of Research

The original research objectives for this project were met. The fiber optic gyroscope's intermittent connectivity was restored. Next, the rate of drift was characterized. Remarkably, the rate of drift is much higher (1.7 degrees per minute) than the manufacturer originally denoted. This in turn, could impact the calculated MOI in this research. The yaw angle requires accurate measurements in order to accurately calculate MOI based upon yaw orientation angle and reaction wheel speeds.

Next, the reaction wheel math model was formulated. Using the data obtained in the characterization of the reaction wheel system, the baseline MOI of SimSat was calculated. Unfortunately, the reaction wheel model is not identical to the observed hardware. Yet, it provides an adequate representation of the hardware when adjusted for a specific level input command, and is usable in the more advanced models in this research. Next, the simplified closed-loop SimSat model was created as a truth source for this research. This allowed a comparison between the baseline model of SimSat and the more advanced models. Finally, several more complex closed-loop math models of SimSat were created. These models ranged from a simple SimSat plant with a PD controller, to a combined PD/RW model, and concluded with the initial development of an advanced satellite dynamic SimSat model. The SimSat/PD controller model and the PD/RW model were extremely accurate for the smaller orientation angle changes when compared to the hardware data. These same models were not accurate for the larger orientation changes. This could be due to the rate limiting that appeared in the 90° run. In addition to the large orientation change model, the satellite dynamic model was unable to be completed due to an unknown error. Despite the fact the model was not completed, the other closed-loop models (specifically the PD/RW model) are representative of the SimSat hardware through small orientation changes.

Using the PD/RW model and the other methodologies developed in this research, the ability to determine the change in fuel using measured MOI was validated using the SimSat hardware. While not able to characterize small changes in the fuel payload (< 1 kg total mass), the techniques used in this research were able to detect a 2 kg mass change from a baseline configuration. However, with further refinement in the models, this technique should be capable of detecting changes smaller than 1 kg.

5.3 Recommendations for Future Research

The following recommendations for future efforts propose three-axis telemetry data analysis, real-time analysis of SimSat’s motions, and a reintegration of SimSat’s systems.

5.3.1 Three-Axis Telemetry Analysis.

Currently, research with SimSat is primarily done with each individual axis. Then, the results for each axis are stated separately or manipulated after the data collection process. It would be faster and more realistic if future efforts concentrated on combined axis maneuvers. These enhanced maneuvers should enhance the ability to determine MOI faster with fewer movements of SimSat.

5.3.2 Dynamic Data Analysis.

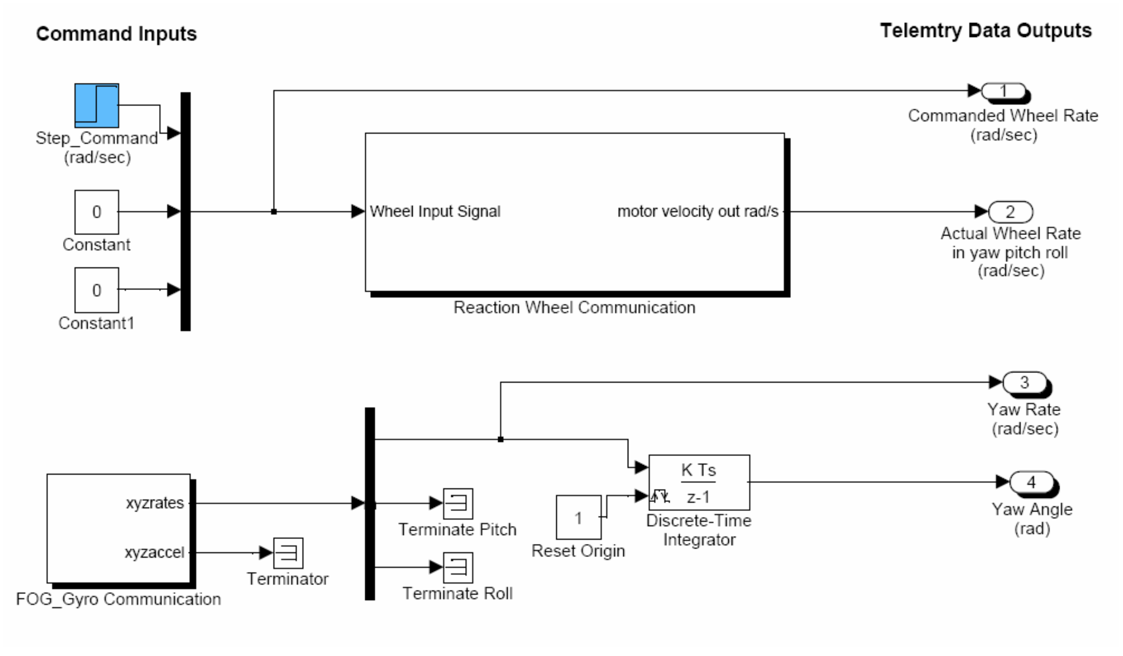
With the completion of the three-axis telemetry analysis, the next step would entail the real time data analysis of the telemetry information. This significant jump from post data analysis to real time analysis would be a substantial improvement to the current system. Additionally, real time analysis has a direct impact for the warfighter; lab tested real time analysis methods can be quickly relayed to Space Command for further testing and possible deployment. The warfighter would have readily accessible technology for immediate use.

5.3.3 Reintegration of SimSat.

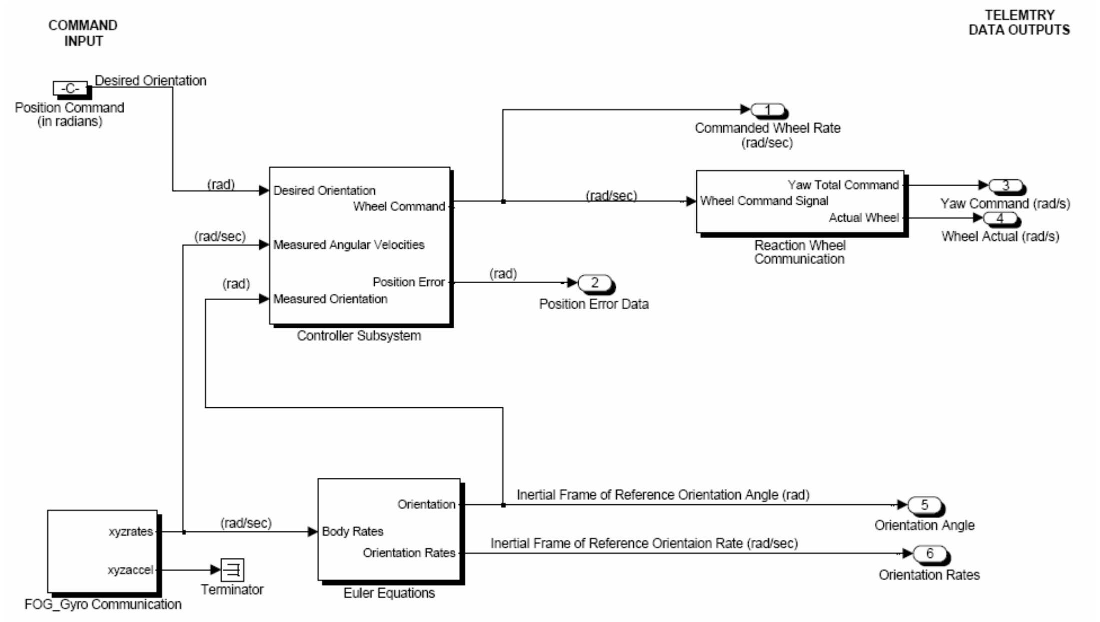
SimSat has had several upgrades since its construction in 1999. However, the installation of the fiber gyro has created a significant issue with SimSat. Over the course of this research, the author suffered numerous delays and corrupted data from the fiber gyro. These delays were due to the gyro failing to perform (give gyro rates every minute) or the gyro providing completely random data (incomprehensible gyro rates). Through troubleshooting, the author found that SimSat did not suffer as many malfunctions when the gyro was operating off of wall power than versus batteries. Because gyro errors continued to occur, the author connected the gyro back to the original Matlab® integration code bypassing dSPACE®; this change led the gyro to suffer no anomalies. This configuration required the gyro to be connected directly to the computer which made SimSat not suitable for operation. The fault tree analysis proves it can either be the power supply or the integration of the fiber gyro with dSPACE®. If SimSat were completely disassembled and reintegrated the compatibility issue with the fiber gyro should be resolved.

In an era of tight government budgets, every pound of satellite fuel must be conserved. By demonstrating the use of MOI as a viable method in determining remaining satellite fuel provides the satellite operator another tool to determine the useful life of the satellite. This enhanced ability allows the maximum use of the satellite life, and therefore extends the ever shrinking budget.

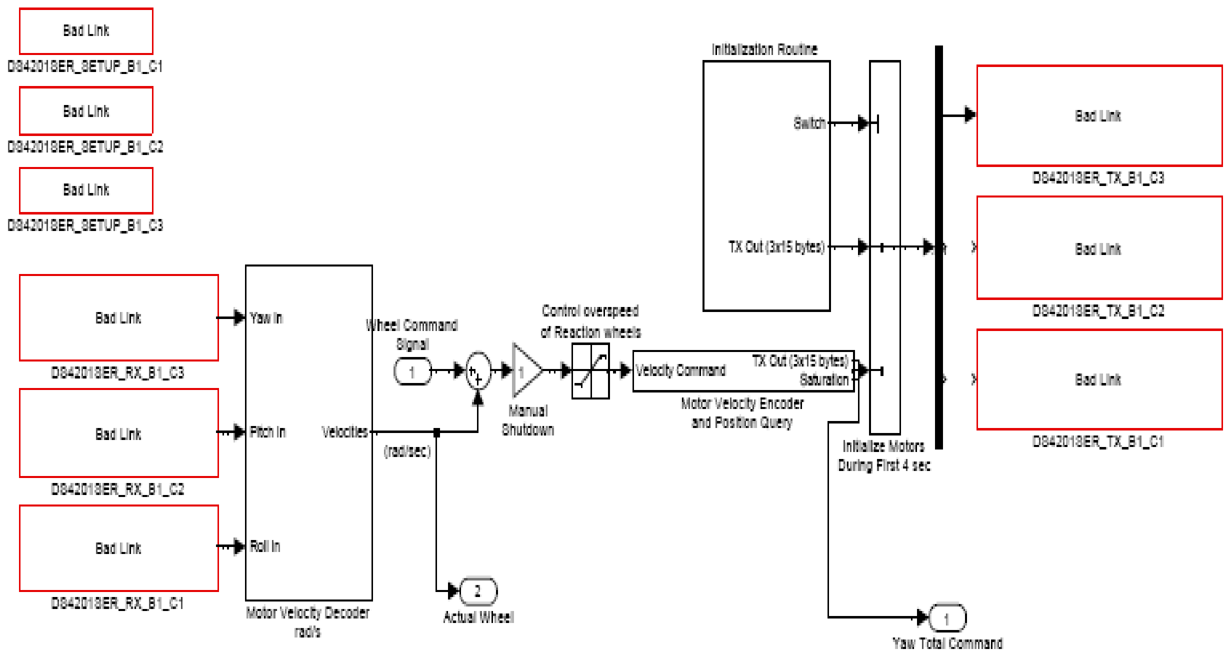
Appendix A: SimSat Hardware Models



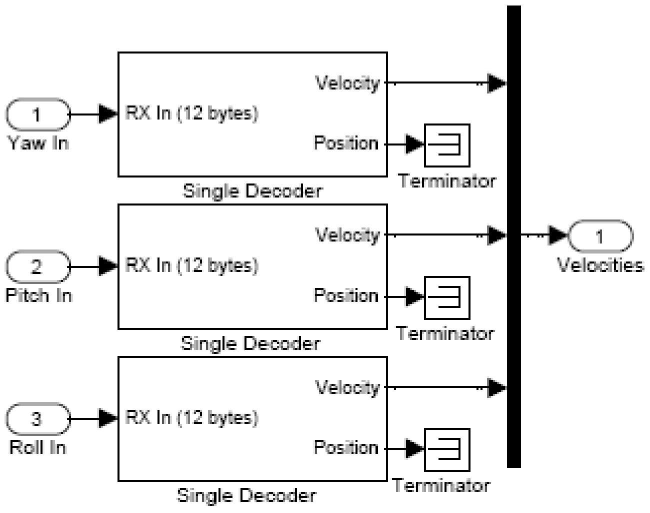
Top Level SimSat Hardware Model Used to Determine Reaction Wheel Response and Baseline MOI Values for SimSat



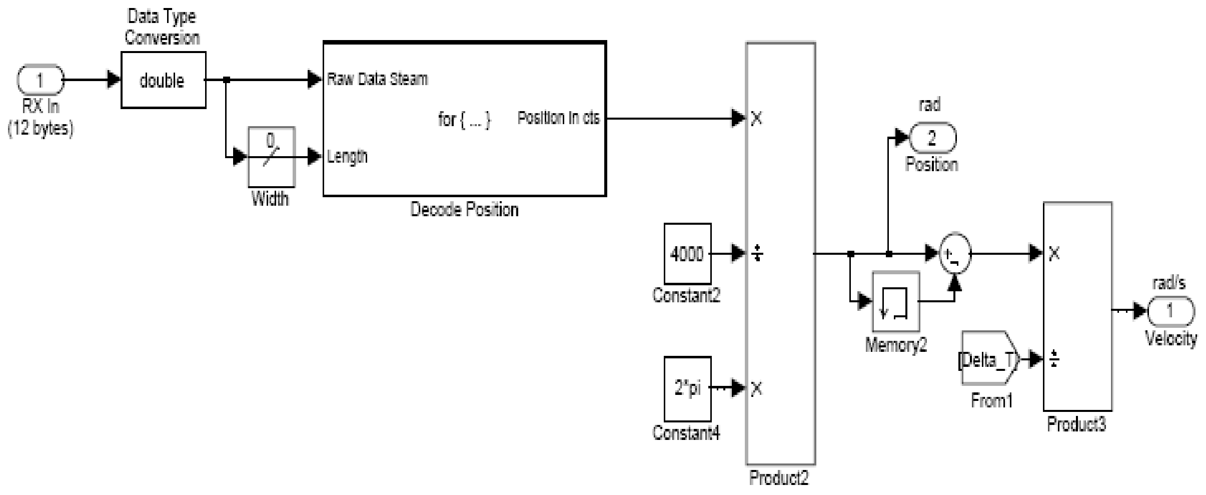
Top Level SimSat Hardware Model Used to Determine Orientation Changes



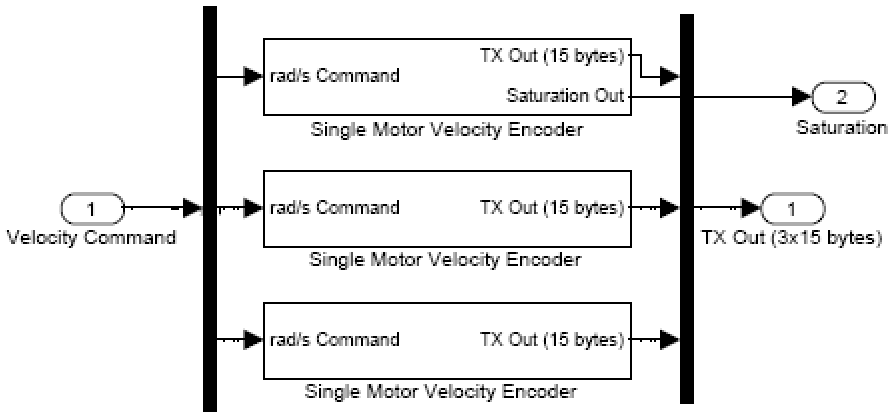
Reaction Wheel Communication Main Block



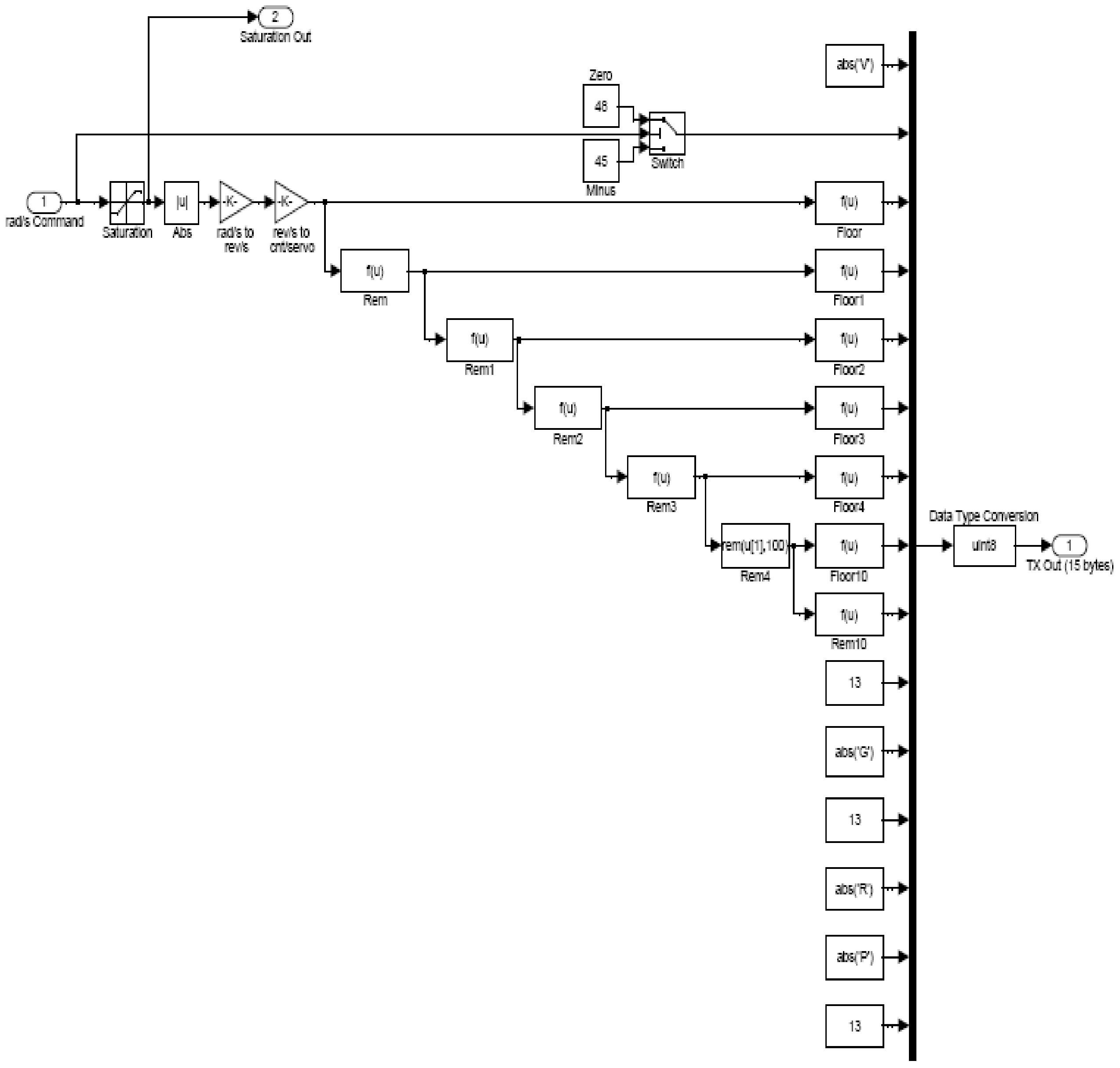
Motor Velocity Decoder



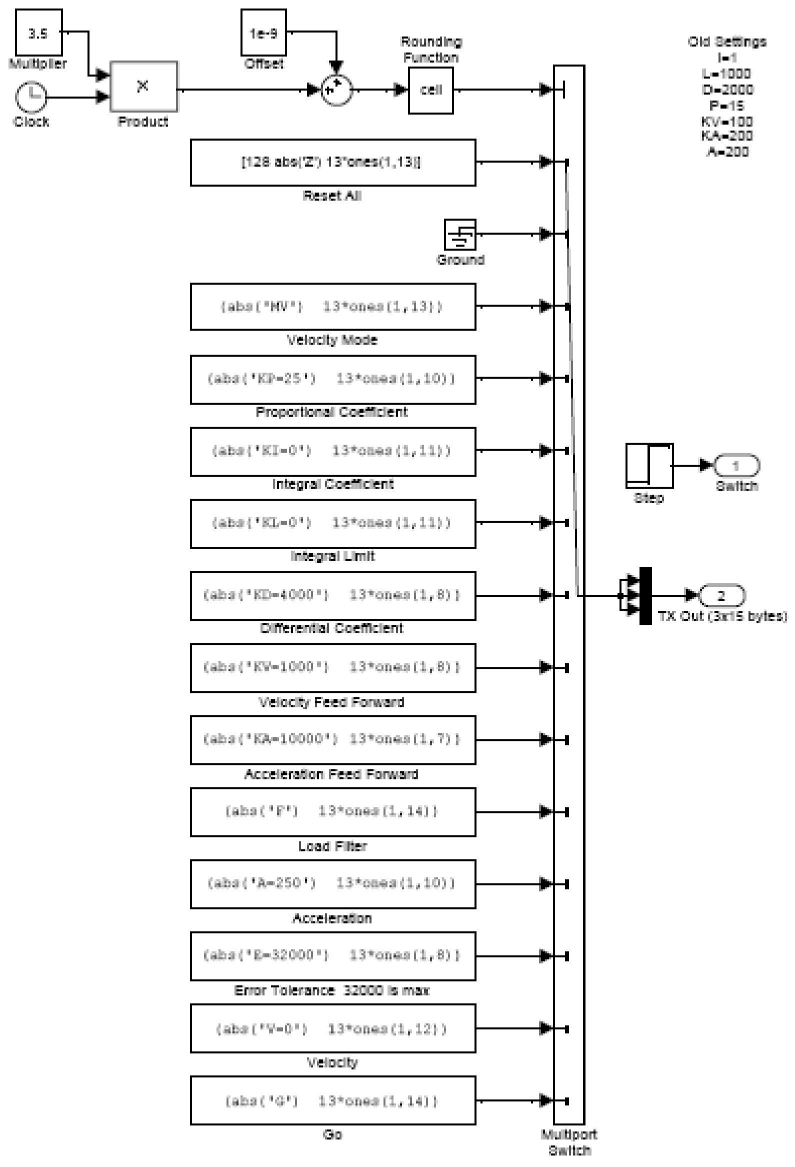
Single Decoder Block from the Motor Velocity Decoder



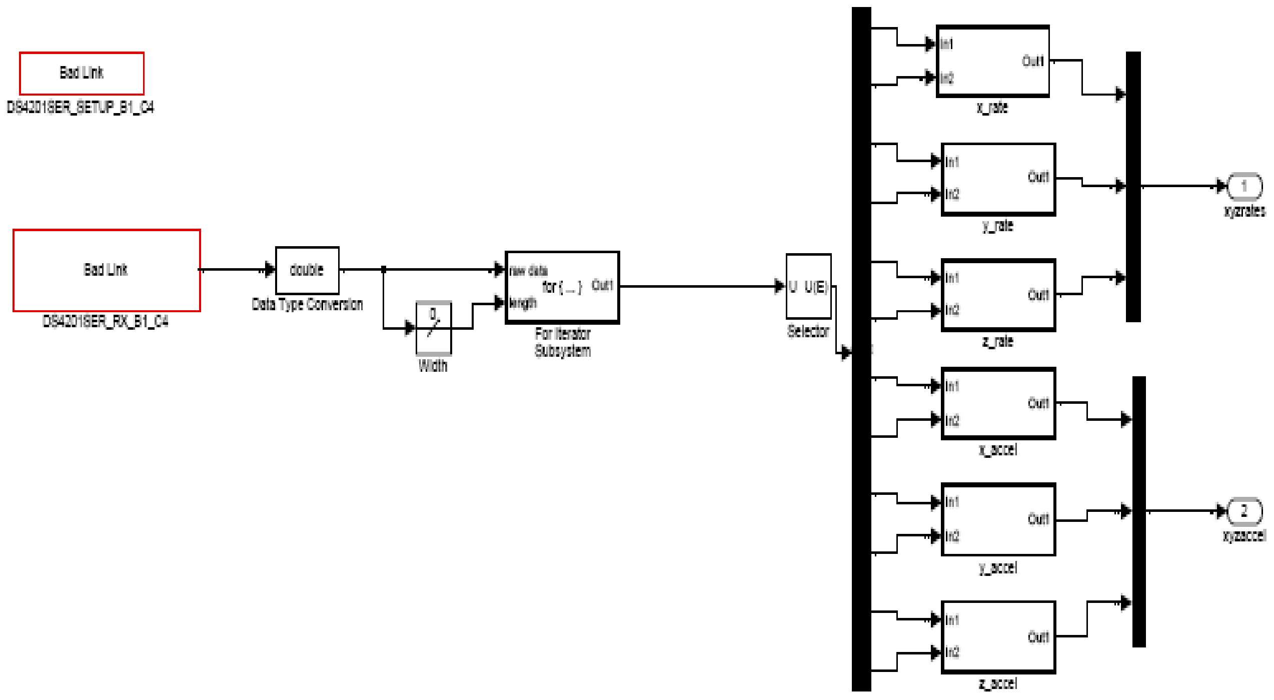
Motor Velocity Encoder



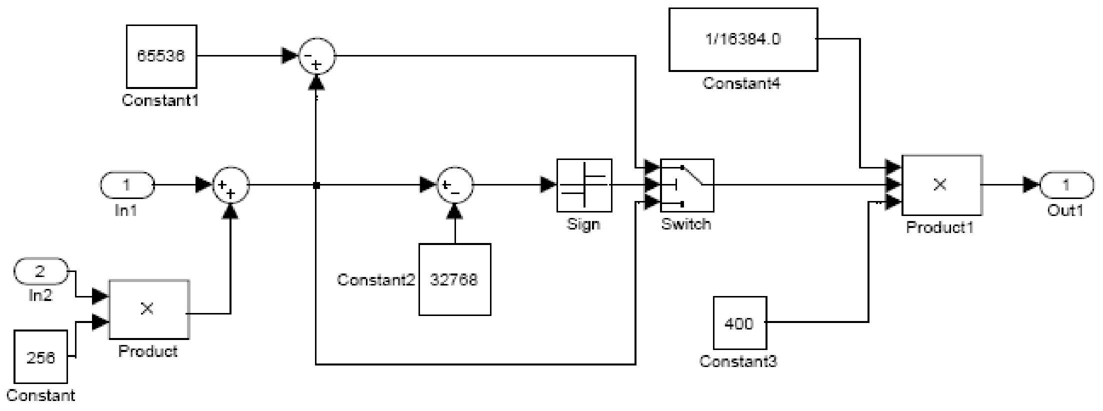
Single Encoder Block from the Motor Velocity Encoder



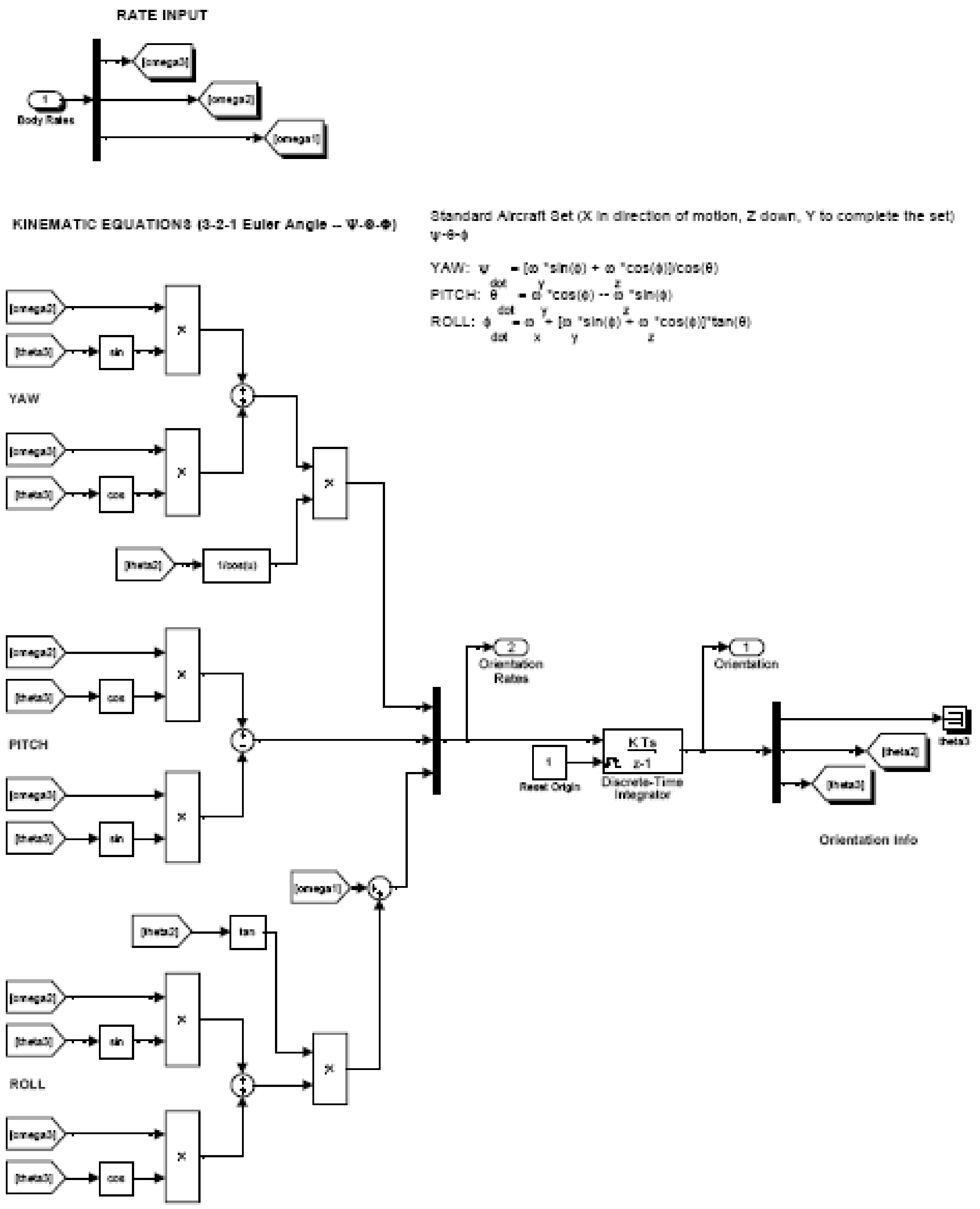
Reaction Wheel Initialization Routine from Reaction Wheel Communication Block



Fiber Optic Gyro Communication Top Level Block



XYZ Rate Block from Fiber Optic Gyro Main Block


RATE INPUT
1
Body Rates
[omega3]
[omega2]
[omega1]
KINEMATIC EQUATIONS (3-2-1 Euler Angle -- Ψ-Θ-Φ)
Standard Aircraft Set (X in direction of motion, Z down, Y to complete the set)
ψ-θ-ϕ
YAW: ψdot = [ωy*sin(ϕ) + ωz*cos(ϕ)]/cos(θ)
PITCH: θdot = ωy*cos(ϕ) -- ωz*sin(ϕ)
ROLL: ϕdot = ωx + [ωy*sin(ϕ) + ωz*cos(ϕ)]*tan(θ)
YAW
PITCH
ROLL
[theta3]
[theta2]
sin
cos
tan
1/cos(u)
x
2
Orientation Rates
1
Orientation
1
Reset Origin
K Ts / z-1
Discrete-Time Integrator
theta3
Orientation Info

Euler Angle Orientation Equations

Appendix B: Determine SimSat MOI in 3 Axis Matlab® Code

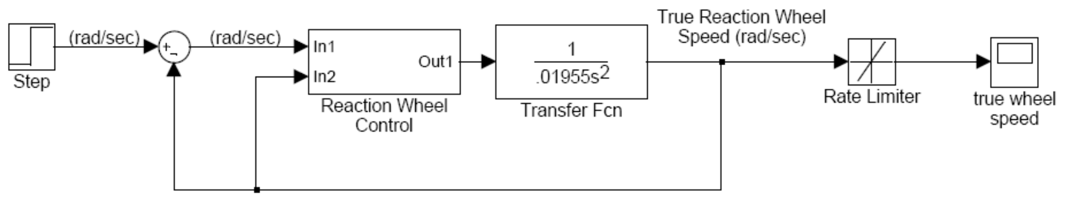
```
% Determine SimSat Moment of Inertia 3 Axis
% Written by Capt Jason Geitgey
% June 06
% Open source as long as credit is given

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This code is designed to generate the MOI for SimSat from the open loop data
% obtained from inputting a step command the reaction wheel in the yaw, pitch,
% or roll axis one at a time.  The yaw method is first and uses the
% constant velocity of SimSat and the reaction wheel to get the MOI.  Have
% to take the files individually to get each MOI.  The pitch and roll axis
% use the change in position and ratios to get the MOI.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% YAW METHOD HERE

clear all; clc; close all;
format long e;

load 5rad_sec_1          %Vary all 1-3 and below in data 1-3 to match what
%load 10rad_sec_1        %# being used here.  Have to match!
%load 15rad_sec_1
%load 20rad_sec_1
%load 25rad_sec_1        %Can only load 1 & 3 #2 data was corrupt for gyro
%load 30rad_sec_1
%load 35rad_sec_1
%load 40rad_sec_1         %45 rad/sec gyro data bad
%load 50rad_sec_1         %50 rad/sec gyro data #3 bad
%load 55rad_sec_1
%load 65rad_sec_3         %65 rad/sec gyro data #2 and #3 bad
                          %All other gyro data is bad

% Generate plots showing the actual yaw wheel speed "w_rwheel" (rad/sec) vs. the
% resulting actual SimSat yaw speed "w_simsat" (rad/sec)
% Used in theory section and helpful to make sure steady state velocities
% are being used to determine the MOI of SimSat

data = rad_sec_3
t = data.X.Data';        % time increment for data collected in .05 step increments due to model settings
num_pts = length(t);
w_rwheel = data.Y(1,1).Data';       % wheel speed actual from SimSat (rad/sec)
wheel_cmd = data.Y(1,2).Data';    % wheel speed commanded on SimSat (rad/sec)
yaw_ang = data.Y(1,3).Data';        % SimSat yaw angle (rad)
w_simsat = data.Y(1,4).Data';        % SimSat yaw rate (rad/sec)

figure(1)
plot(t, w_rwheel, '-r', t, w_simsat*100, '-b')
grid on
title('Comparison of actual wheel velocity vs. resulting SimSat velocity scaled')
xlabel('Time (sec)')
ylabel('Velocity (rad/sec)')
legend('w_r_w_h_e_e_l','w_s_i_m_s_a_t*100')
```

```

% Generate a plot using the same info as above just using the absolute
% values of the collected data

figure(2)
plot(t, abs(w_rwheel), '-r', t, abs(w_simsat*100), '-b')
grid on
title('Comparison of absolute values of wheel velocity vs. SimSat velocity scaled')
xlabel('Time (sec)')
ylabel('Velocity (rad/sec)')
legend('w_r_w_h_e_e_l','w_s_i_m_s_a_t*100')


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Code used below is used to find the MOI of SimSat from the obtained
% hardware data from above.  CAUTION  You must be careful to only use the
% steady state numbers obtained for SimSat and the reaction wheel as they
% rotate at constant velocities in the yaw direction.  Can be
% used on any of the above data sets but watch the time settings that are used.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Moment of inertia of the reaction wheel determined from previous
% experiments and based on hardware in SimSat lab

I_rw = 1.955099417802845e-002;      % Units are (kg*m^2)

w_rw_mean = mean(w_rwheel(141:401))
w_sat_mean = abs(mean(w_simsat(141:401)))
I_sat = (I_rw*(w_rw_mean-w_sat_mean))/w_sat_mean

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Use the running total positions of the reaction wheel and SimSat in the
% pitch and roll direction in order to calculate the MOI in those
% directions.  The constant velocity approach will not work due to gravity
% effects.  Once the wheel stops accelerating and reaches constant speed
% there is no longer a torque and SimSat has gravity pull it back to an
% initial condition.  Once the plots are generated can determine slope and
% get the MOI through ratios.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% PITCH METHOD BELOW

load pitch_ol_5_rad_s               %Vary the file loaded just as above in
%load pitch_ol_5_rad_s2             %order to get the file to run.  The
%load pitch_ol_5_rad_s3             %file callout here must match below.
%load pitch_ol_10_rad_s
%load pitch_ol_10_rad_s2
%load pitch_ol_10_rad_s3
%load pitch_ol_15_rad_s
%load pitch_ol_15_rad_s2
%load pitch_ol_15_rad_s3

t2 = pitch_ol_5_rad_s.X.Data';
num_pts2 = length(t2);
```

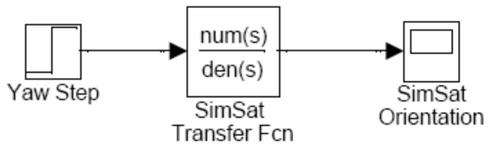
```
pitch_rwh = pitch_ol_5_rad_s.Y(1,2).Data';
pitch_ang = pitch_ol_5_rad_s.Y(1,4).Data';
pitch_rate = pitch_ol_5_rad_s.Y(1,5).Data';

for i = 1:num_pts2;
    init = i;
    final = i+1;
    if init < num_pts2;
        rw_change(i) = pitch_rwh(final) - pitch_rwh(init);
        pitch_change(i) = pitch_ang(final) - pitch_ang(init);
    else init == num_pts2;
        rw_change(i) = pitch_rwh(init) - pitch_rwh(init);
        pitch_change(i) = pitch_ang(init) - pitch_ang(init);
    end

    i = i+1
end
rad_rw_change = rw_change * .05;
rad_rw_change = abs(rad_rw_change);
pitch_chg = abs(pitch_change)
for n = 1:num_pts2;
    start = n;
    finish = n-1;
    if n == 1;
        Pos_rwheel(n) = rad_rw_change(n);
        Pos_pitch(n) = pitch_chg(n);
    else n > 1;
        Pos_rwheel(n) = Pos_rwheel(finish) + rad_rw_change(start);
        Pos_pitch(n) = Pos_pitch(finish) + pitch_chg(start);
    end
    n = n+1;
end

figure(3)
plot(t2, Pos_rwheel, '-b', t2, 10*Pos_pitch, '-r')
grid on
title('Comparison of position change in pitch reaction wheel vs. SimSat')
xlabel('Time (sec)')
ylabel('Position (rad)')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% ROLL METHOD BELOW

load roll_ol_5_rad_s      %Have to match file loaded and variables below
%load roll_ol_5_rad_s2
%load roll_ol_5_rad_s3
%load roll_ol_10_rad_s
%load roll_ol_10_rad_s2
%load roll_ol_10_rad_s3

t3 = roll_ol_5_rad_s.X.Data';
num_pts3 = length(t3);
roll_rwh = roll_ol_5_rad_s.Y(1,3).Data';
roll_ang = roll_ol_5_rad_s.Y(1,5).Data';
```

```
roll_rate = roll_ol_5_rad_s.Y(1,6).Data';

for i = 1:num_pts3;
   init = i;
   final = i+1;
   if init < num_pts3;
      roll_rw_change(i) = roll_rwh(final) - roll_rwh(init);
      roll_change(i) = roll_ang(final) - roll_ang(init);
   else init == num_pts3;
      roll_rw_change(i) = roll_rwh(init) - roll_rwh(init);
      roll_change(i) = roll_ang(init) - roll_ang(init);
   end

   i = i+1
end
roll_rad_rw_change = roll_rw_change * .05;
roll_rad_rw_change = abs(roll_rad_rw_change);
roll_chg = abs(roll_change)
for n = 1:num_pts3;
   start = n;
   finish = n-1;
   if n == 1;
      roll_Pos_rwheel(n) = roll_rad_rw_change(n);
      roll_Pos_pitch(n) = roll_chg(n);
   else n > 1;
      roll_Pos_rwheel(n) = roll_Pos_rwheel(finish) + roll_rad_rw_change(start);
      roll_Pos_pitch(n) = roll_Pos_pitch(finish) + roll_chg(start);
   end
   n = n+1;
end

figure(4)
plot(t3, roll_Pos_rwheel, '-b', t3, 10*roll_Pos_pitch, '-r')
grid on
title('Comparison of position change in roll reaction wheel vs. SimSat')
xlabel('Time (sec)')
ylabel('Position (rad)')


%End of File
```

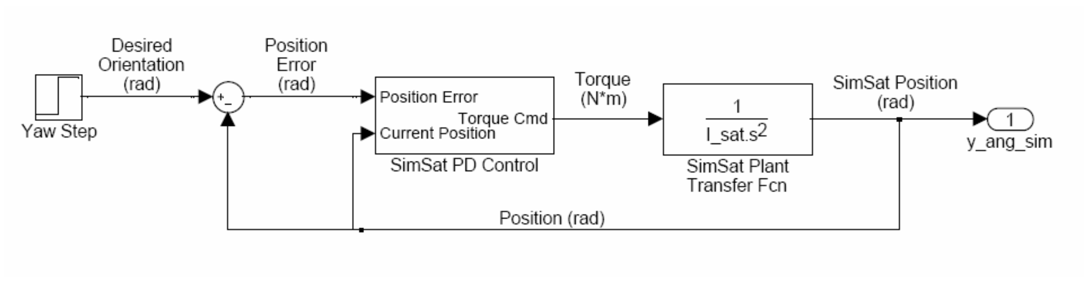
Appendix C: Mathematical Models



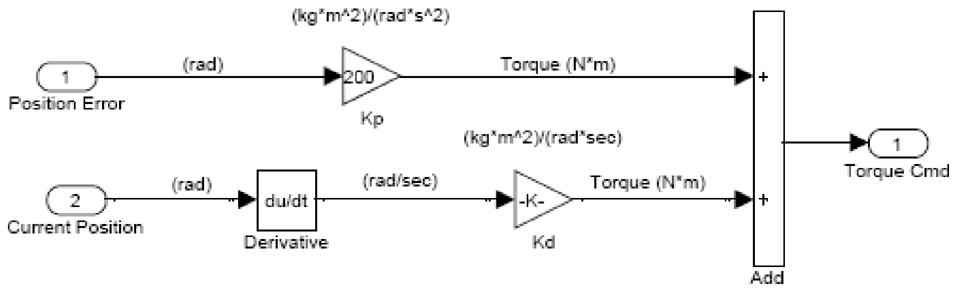
Reaction Wheel Math Model Top Level



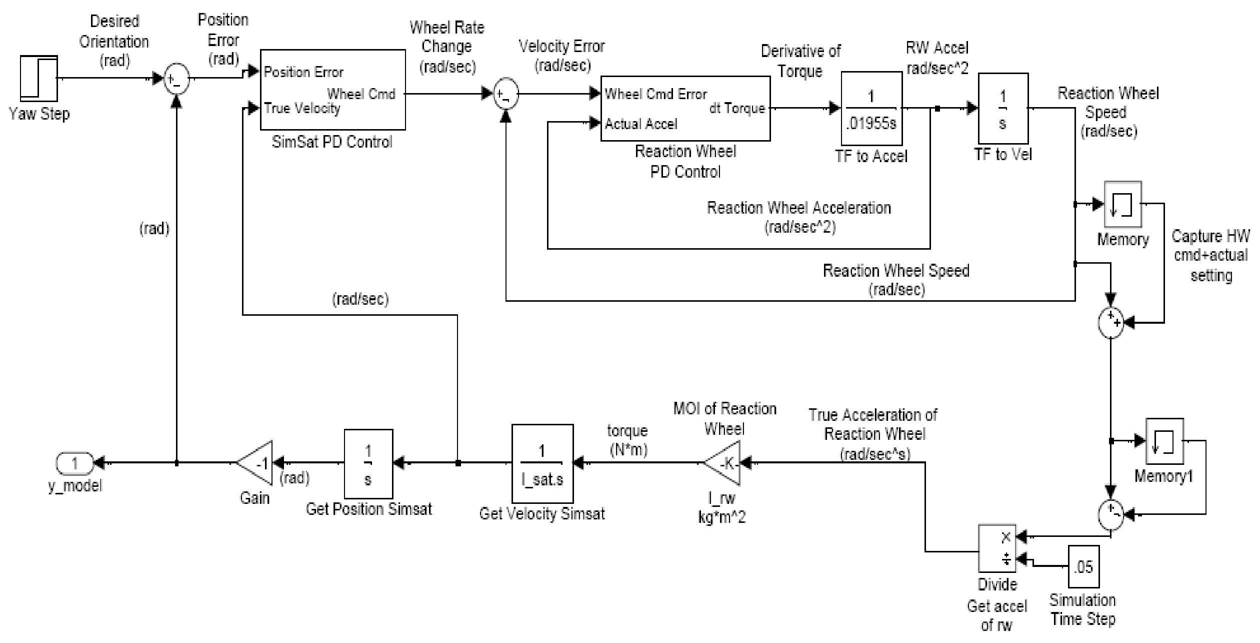
Simplified Closed-Loop SimSat Math Model



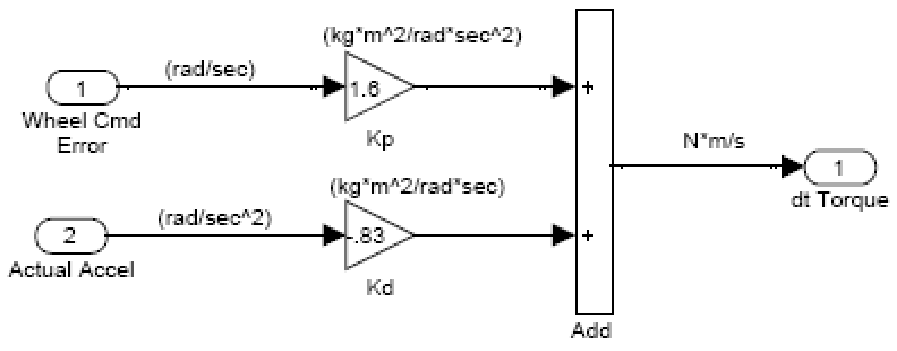
PD/SimSat Math Model Top Level



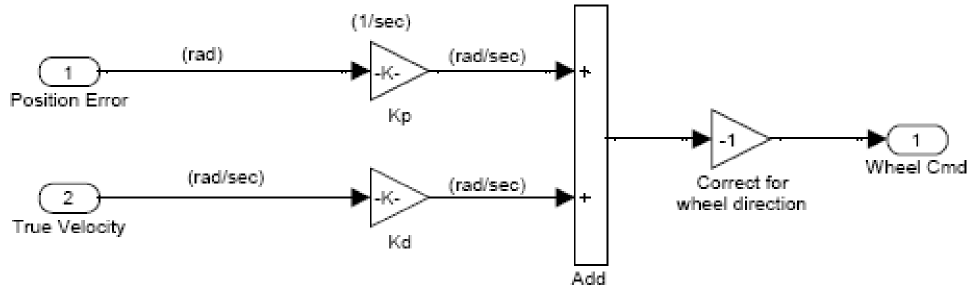
SimSat PD Control Block for PD/SimSat Math Model



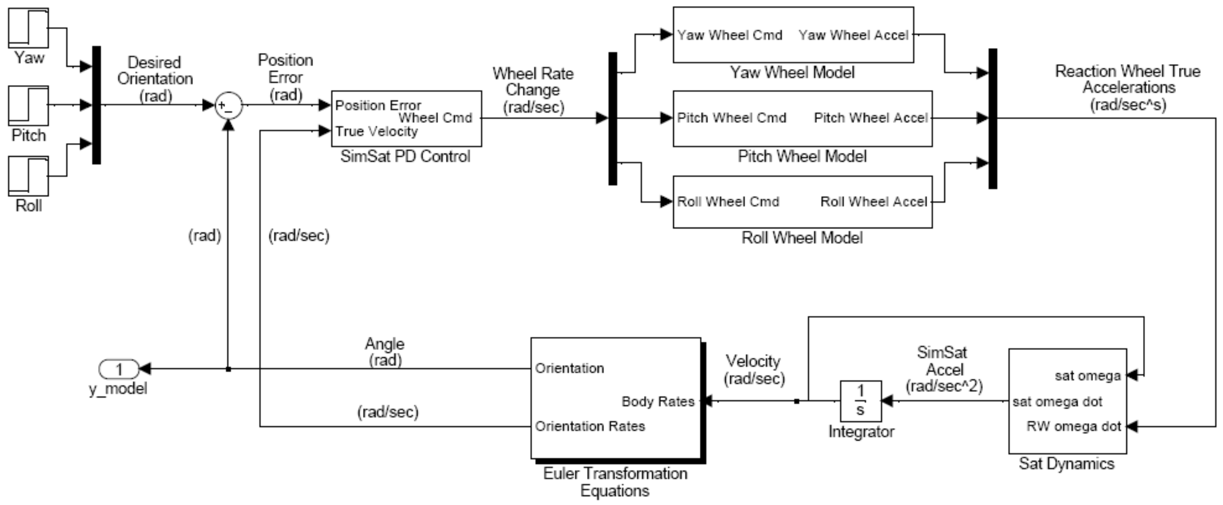
Proportional Derivative and Reaction Wheel SimSat Math Model Top Level



Reaction Wheel PD Control Block for PD/RW Model



SimSat PD Control Block for PD/RW Model



Satellite Dynamic SimSat Math Model

Appendix D: Reaction Wheel Commanded vs Actual Plot Generator Matlab® Code

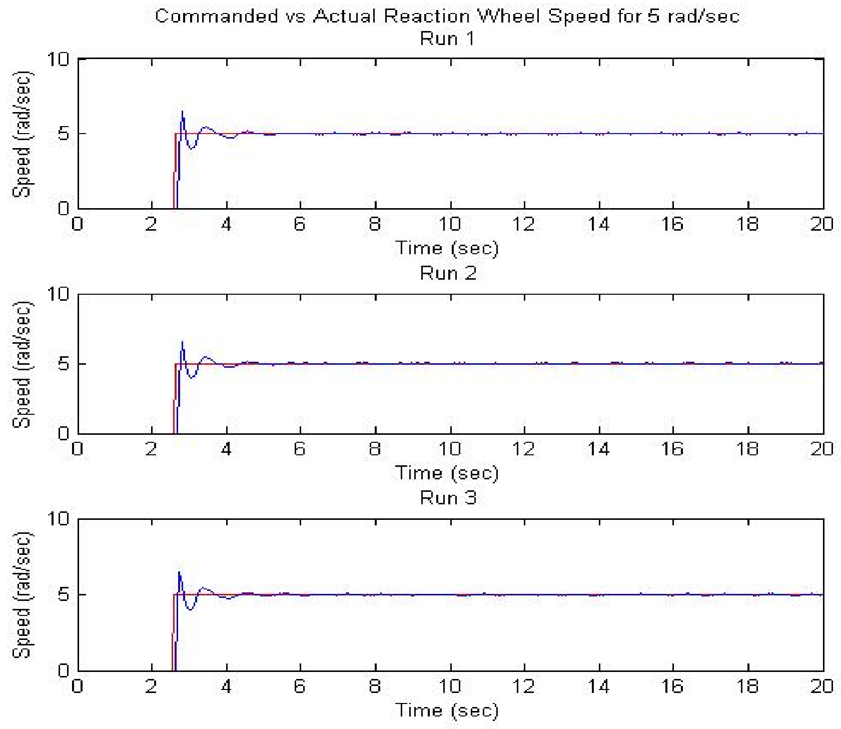
```
% Compare wheel speed vs time
% Capt Jason Geitgey
% June 06
% Open source as long as credit is given
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This code takes data obtained from the SimSat hardware and plots the
% open loop yaw reaction wheel speed vs a commanded step input.  These
% plots are then used to generate the open loop math model for the reaction
% wheel system.  Not all of the plots will start at the same time due to
% variances in the SimSat hardware.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all; clc; close all;

% Select the amount of data you want to plot using this routine by varying
% the i constraint as identified below.

for i = 1:16

    if i == 1
        load 5rad_sec_1
        load 5rad_sec_2
        load 5rad_sec_3
    elseif i == 2
        load 10rad_sec_1
        load 10rad_sec_2
        load 10rad_sec_3
    elseif i == 3
        load 15rad_sec_1
        load 15rad_sec_2
        load 15rad_sec_3
    elseif i == 4
        load 20rad_sec_1
        load 20rad_sec_2
        load 20rad_sec_3
    elseif i == 5
        load 25rad_sec_1
        load 25rad_sec_2
        load 25rad_sec_3
    elseif i == 6
        load 30rad_sec_1
        load 30rad_sec_2
        load 30rad_sec_3
    elseif i == 7
        load 35rad_sec_1
        load 35rad_sec_2
        load 35rad_sec_3
    elseif i == 8
        load 40rad_sec_1
        load 40rad_sec_2
        load 40rad_sec_3
    elseif i == 9
```

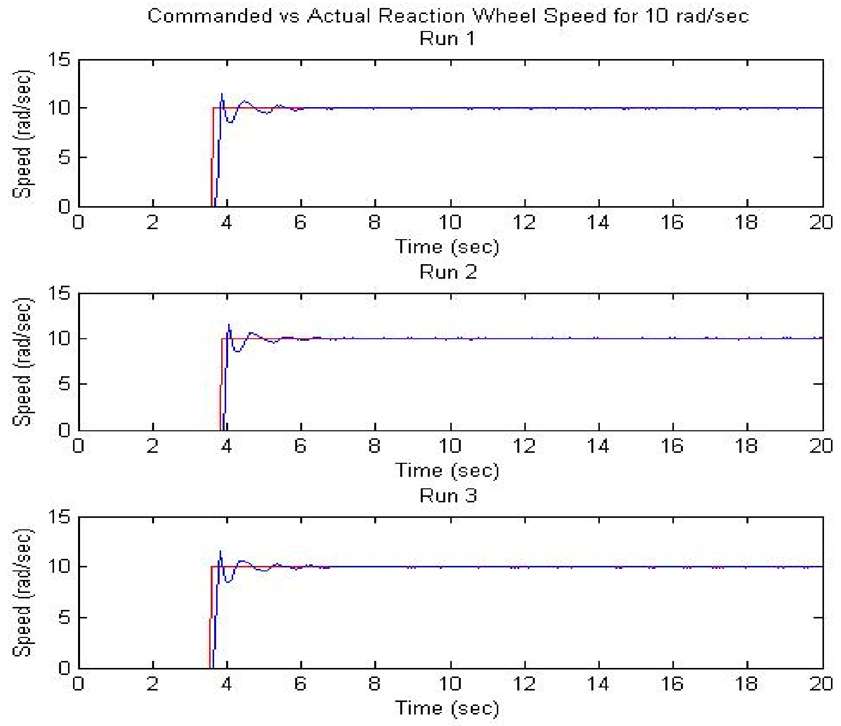
```
        load 45rad_sec_1
        load 45rad_sec_2
        load 45rad_sec_3
    elseif i == 10
        load 50rad_sec_1
        load 50rad_sec_2
        load 50rad_sec_3
    elseif i == 11
        load 55rad_sec_1
        load 55rad_sec_2
        load 55rad_sec_3
    elseif i == 12
        load 60rad_sec_1
        load 60rad_sec_2
        load 60rad_sec_3
    elseif i == 13
        load 65rad_sec_1
        load 65rad_sec_2
        load 65rad_sec_3
    elseif i == 14
        load 70rad_sec_1
        load 70rad_sec_2
        load 70rad_sec_3
    elseif i == 15
        load 75rad_sec_1
        load 75rad_sec_2
        load 75rad_sec_3
    elseif i == 16
        load 80rad_sec_1
        load 80rad_sec_2
        load 80rad_sec_3
    end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Code for plotting the data loaded into the workspace from above
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    for n = 1:3
        if n == 1
            data1 = rad_sec_1
            t1 = data1.X.Data';             % time increment for data collected in .05
                                            % step increments due to model settings
            wheel_act1 = data1.Y(1,1).Data';          % wheel speed actual from SIMSAT (rad/sec)
            wheel_cmd1 = data1.Y(1,2).Data';          % wheel speed commanded on SIMSAT (rad/sec)
            yaw_ang1 = data1.Y(1,3).Data';            % SIMSAT yaw angle (rad)
            yaw_rate1 = data1.Y(1,4).Data';           % SIMSAT yaw rate (rad/sec)
            n = n+1

        elseif n == 2
            data2 = rad_sec_2
            t2 = data2.X.Data';             % time increment for data collected in .05
                                            % step increments due to model settings
            wheel_act2 = data2.Y(1,1).Data';          % wheel speed actual from SIMSAT (rad/sec)
            wheel_cmd2 = data2.Y(1,2).Data';          % wheel speed commanded on SIMSAT (rad/sec)
```

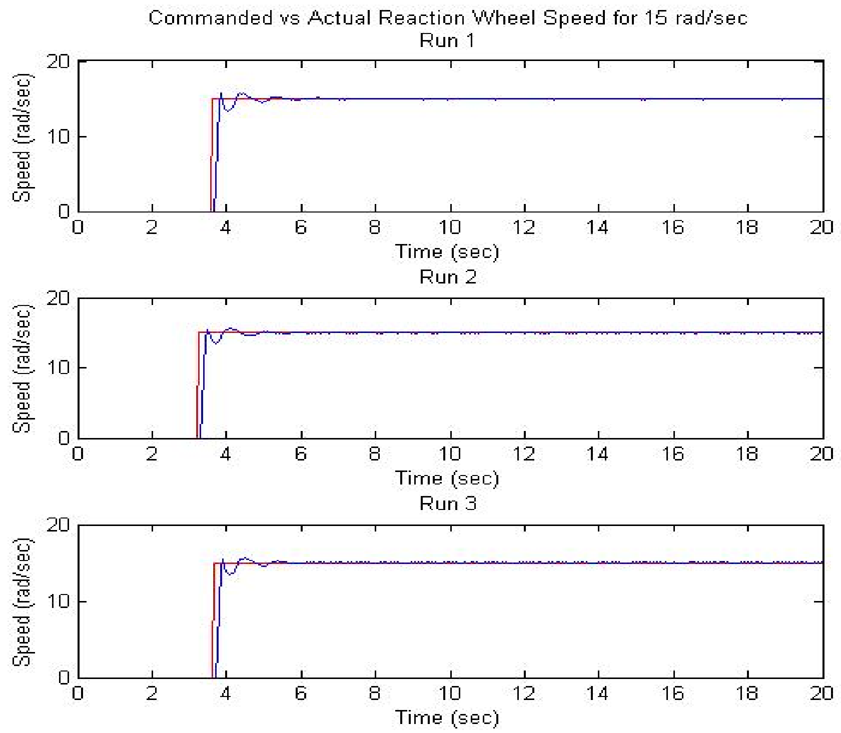
```
          yaw_ang2 = data2.Y(1,3).Data';            % SIMSAT yaw angle (rad)
          yaw_rate2 = data2.Y(1,4).Data';           % SIMSAT yaw rate (rad/sec)
          n = n+1

       elseif n == 3
          data3 = rad_sec_3
          t3 = data3.X.Data';             % time increment for data collected in .05
                                          % step increments due to model settings
          wheel_act3 = data3.Y(1,1).Data';          % wheel speed actual from SIMSAT (rad/sec)
          wheel_cmd3 = data3.Y(1,2).Data';          % wheel speed commanded on SIMSAT (rad/sec)
          yaw_ang3 = data3.Y(1,3).Data';            % SIMSAT yaw angle (rad)
          yaw_rate3 = data3.Y(1,4).Data';           % SIMSAT yaw rate (rad/sec)

       end
    end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate the plots for the currently called set of data.  These plots are
% of the three collected sets of data from SIMSAT for the given input value
% for the wheel speed in rad/sec
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure(i)
subplot(3,1,1);
plot(t1, wheel_cmd1, '-r', t1, wheel_act1);
xlabel('Time (sec)');
ylabel('Speed (rad/sec)');
    if i==1
          title({'Commanded vs Actual Reaction Wheel Speed for 5 rad/sec';'Run 1'})
       elseif i==2
          title({'Commanded vs Actual Reaction Wheel Speed for 10 rad/sec';'Run 1'})
       elseif i==3
          title({'Commanded vs Actual Reaction Wheel Speed for 15 rad/sec';'Run 1'})
       elseif i==4
          title({'Commanded vs Actual Reaction Wheel Speed for 20 rad/sec';'Run 1'})
       elseif i==5
          title({'Commanded vs Actual Reaction Wheel Speed for 25 rad/sec';'Run 1'})
       elseif i==6
          title({'Commanded vs Actual Reaction Wheel Speed for 30 rad/sec';'Run 1'})
       elseif i==7
          title({'Commanded vs Actual Reaction Wheel Speed for 35 rad/sec';'Run 1'})
       elseif i==8
          title({'Commanded vs Actual Reaction Wheel Speed for 40 rad/sec';'Run 1'})
       elseif i==9
          title({'Commanded vs Actual Reaction Wheel Speed for 45 rad/sec';'Run 1'})
       elseif i==10
          title({'Commanded vs Actual Reaction Wheel Speed for 50 rad/sec';'Run 1'})
       elseif i==11
          title({'Commanded vs Actual Reaction Wheel Speed for 55 rad/sec';'Run 1'})
       elseif i==12
          title({'Commanded vs Actual Reaction Wheel Speed for 60 rad/sec';'Run 1'})
       elseif i==13
          title({'Commanded vs Actual Reaction Wheel Speed for 65 rad/sec';'Run 1'})
       elseif i==14
          title({'Commanded vs Actual Reaction Wheel Speed for 70 rad/sec';'Run 1'})
```

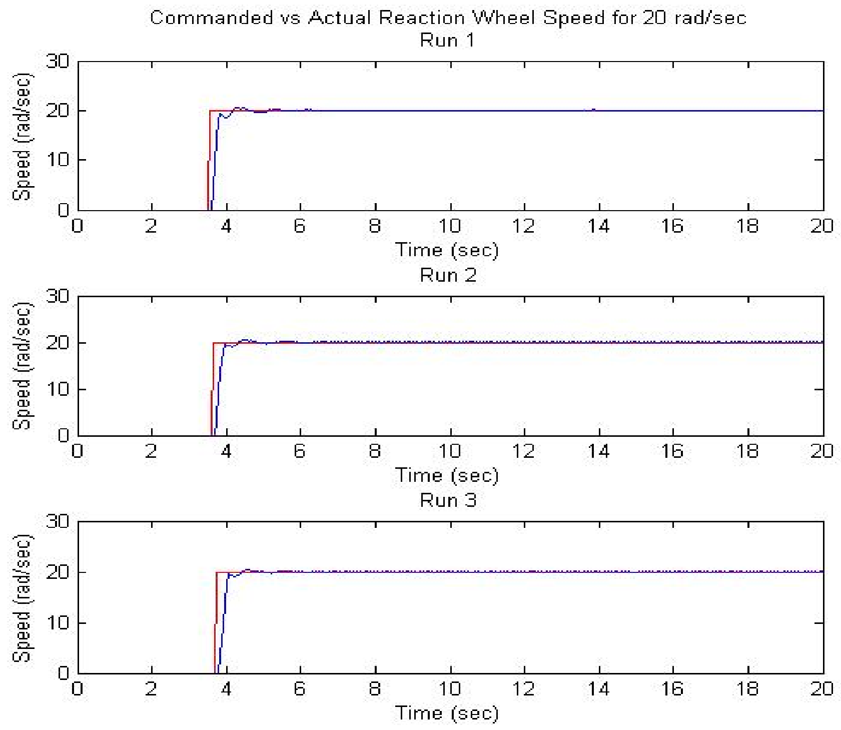
```
        elseif i==15
            title({'Commanded vs Actual Reaction Wheel Speed for 75 rad/sec';'Run 1'})
        elseif i==16
            title({'Commanded vs Actual Reaction Wheel Speed for 80 rad/sec';'Run 1'})
    end

subplot(3,1,2);
plot(t2, wheel_cmd2, '-r', t2, wheel_act2);
xlabel('Time (sec)');
ylabel('Speed (rad/sec)');
title('Run 2');
subplot(3,1,3);
plot(t3, wheel_cmd3, '-r', t2, wheel_act3);
xlabel('Time (sec)');
ylabel('Speed (rad/sec)');
title('Run 3');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This plots just one set of reaction wheel data vs time.  Just select the
% wheel command and wheel actual you want for the selected data counter
% from above.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(i+1)
plot(t1, wheel_cmd1, '-r', t1, wheel_act1, '-b')
xlabel('Time (sec)');
ylabel('Reaction Wheel Speed (rad/sec)')
legend('Wheel Command','Wheel Actual', 'location','SE')

% This is the counter for the file to load and then complete and plot the
% next set of data variables

i = i+1

end
```

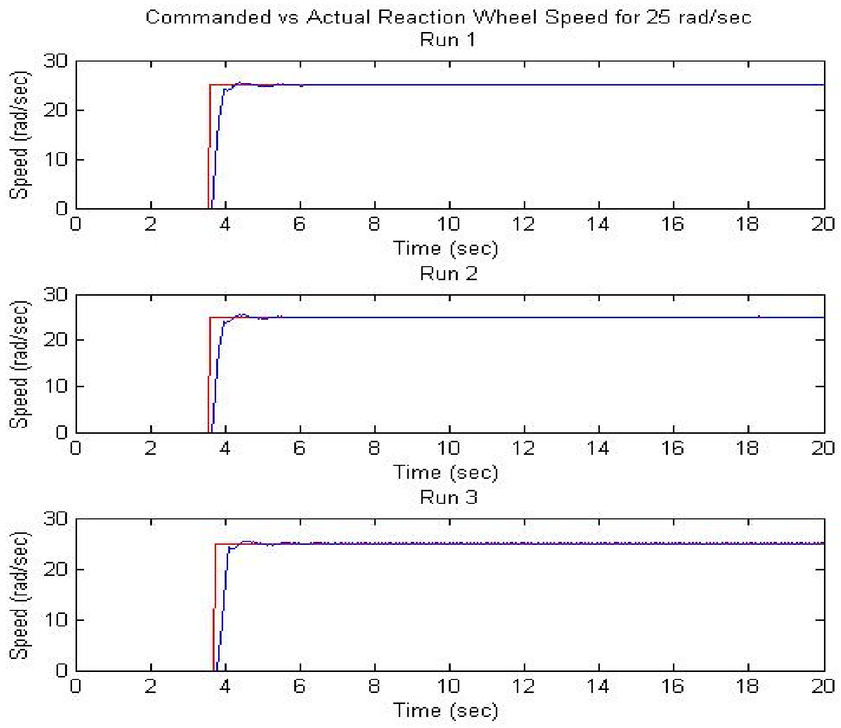
Appendix E: Plots of Commanded vs Actual Reaction Wheel Response

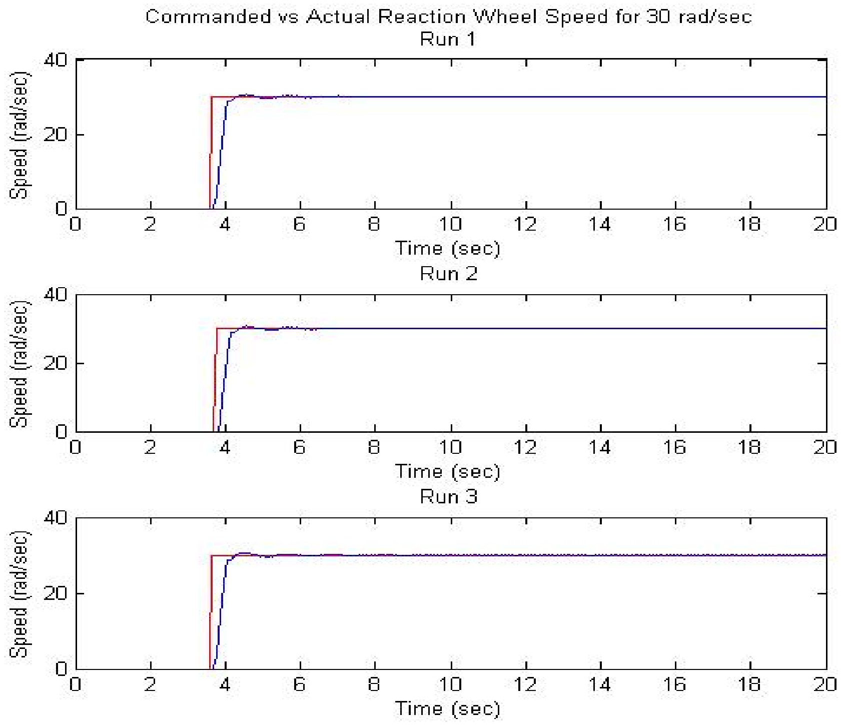


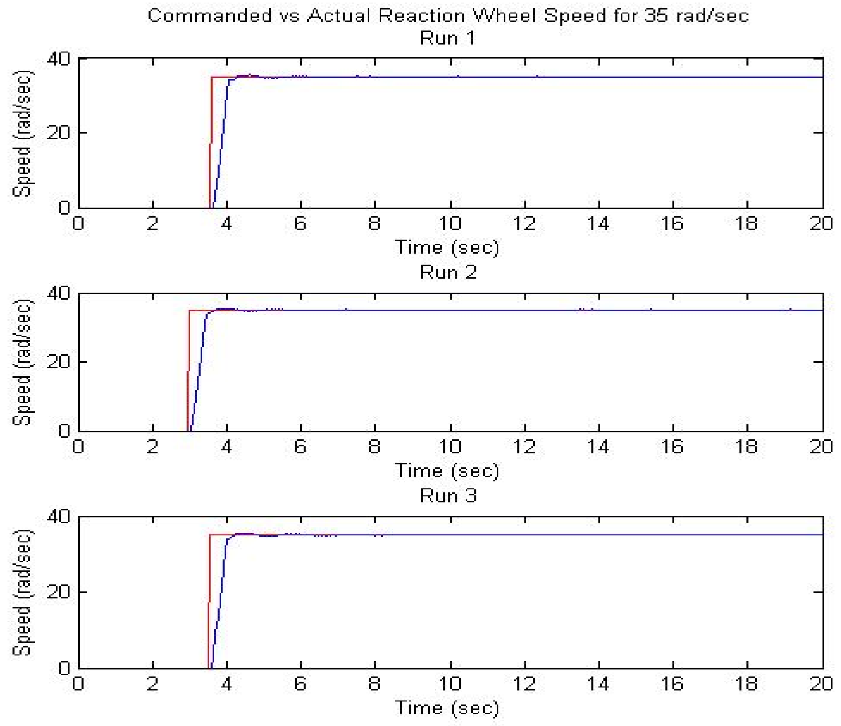


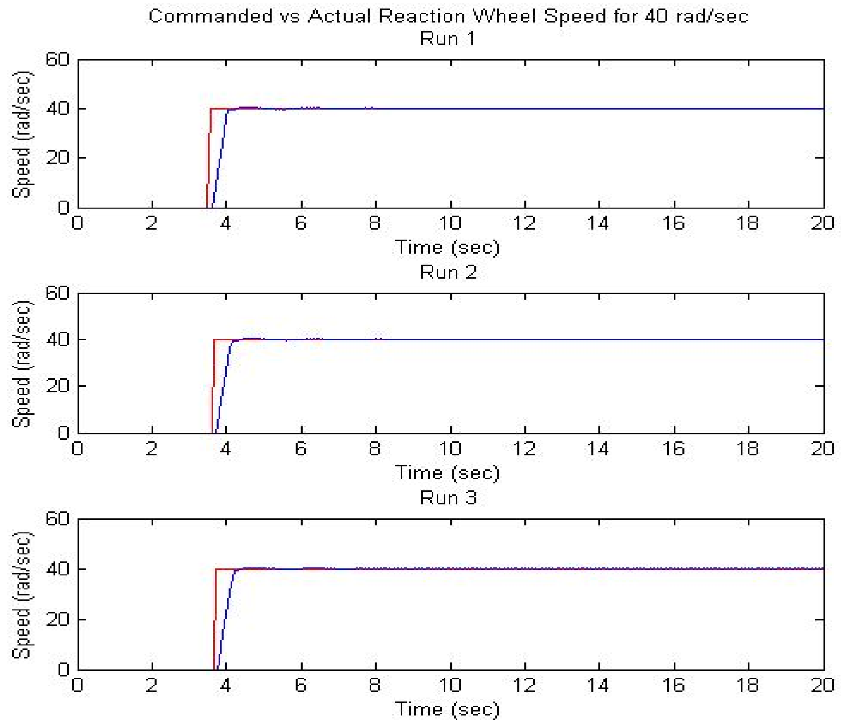

Commanded vs Actual Reaction Wheel Speed for 15 rad/sec
Run 1
Speed (rad/sec)
Time (sec)
Run 2
Speed (rad/sec)
Time (sec)
Run 3
Speed (rad/sec)
Time (sec)
0
10
20
2
4
6
8
12
14
16
18

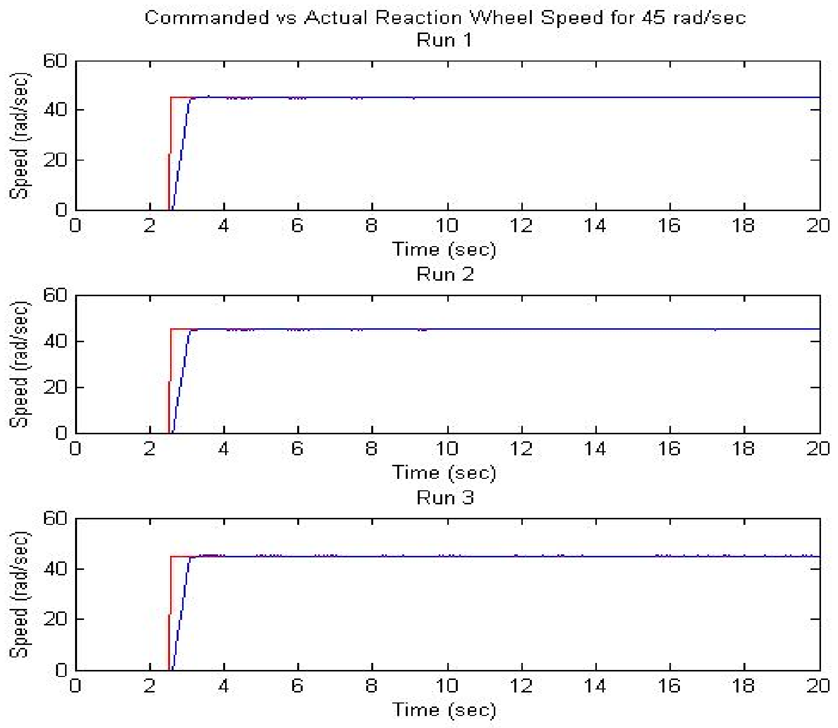

Commanded vs Actual Reaction Wheel Speed for 20 rad/sec
Run 1
Speed (rad/sec)
Time (sec)
Run 2
Speed (rad/sec)
Time (sec)
Run 3
Speed (rad/sec)
Time (sec)
0
10
20
30
2
4
6
8
12
14
16
18

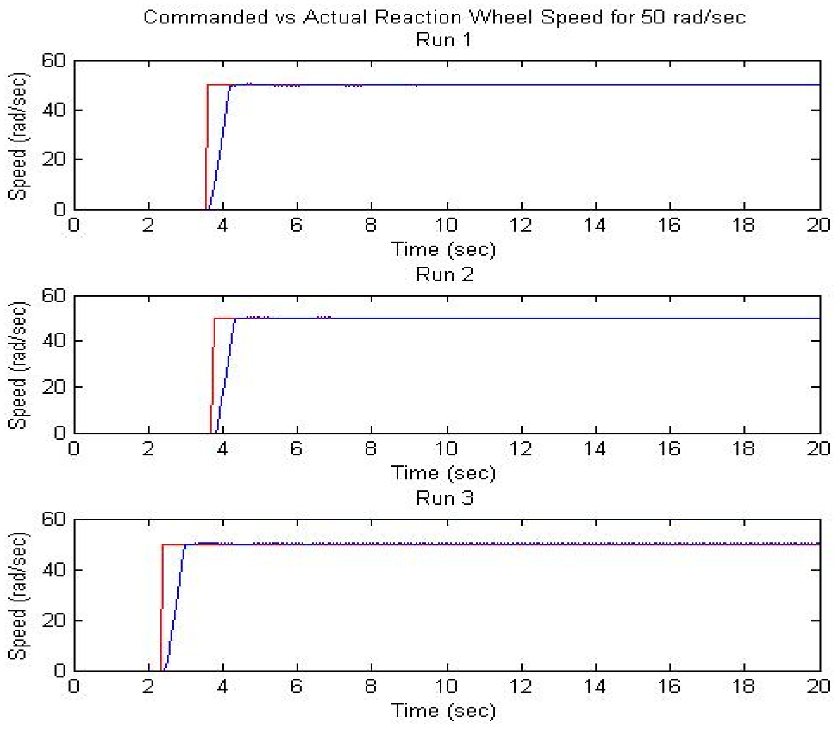

Commanded vs Actual Reaction Wheel Speed for 25 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)

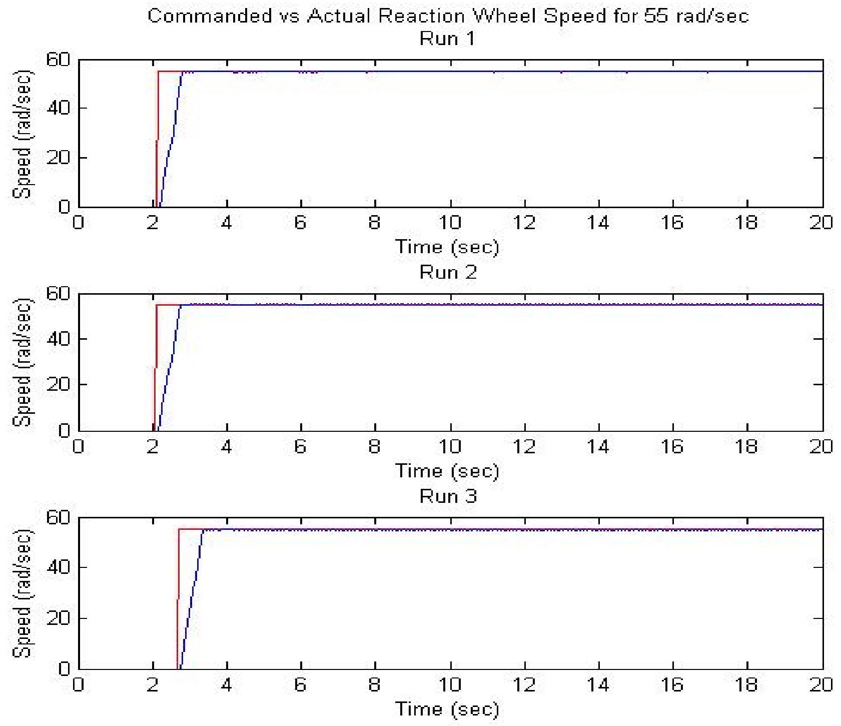

Commanded vs Actual Reaction Wheel Speed for 30 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)

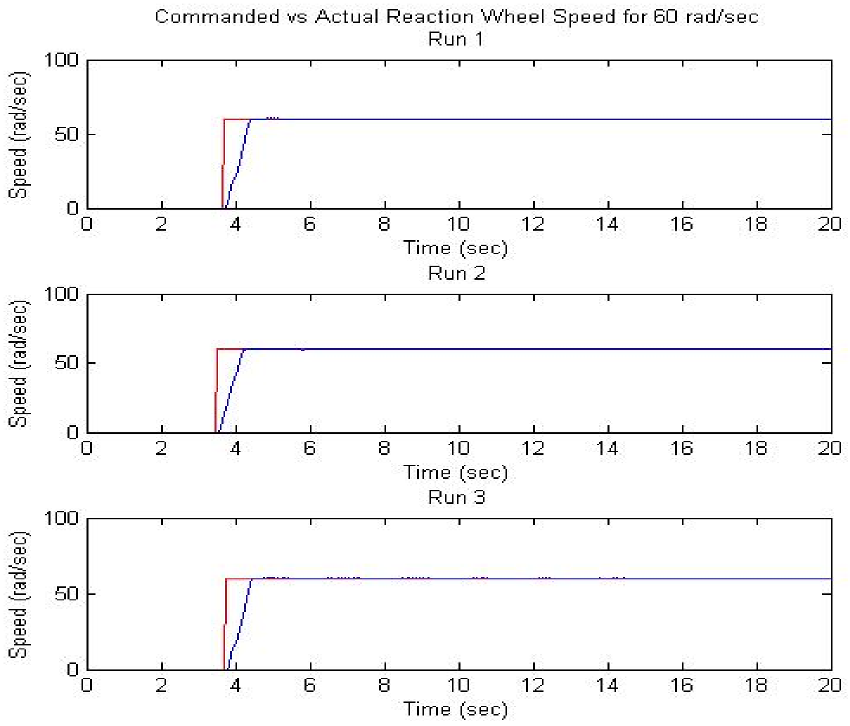

Commanded vs Actual Reaction Wheel Speed for 35 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)
0
2
4
6
8
10
12
14
16
18
20
20
40

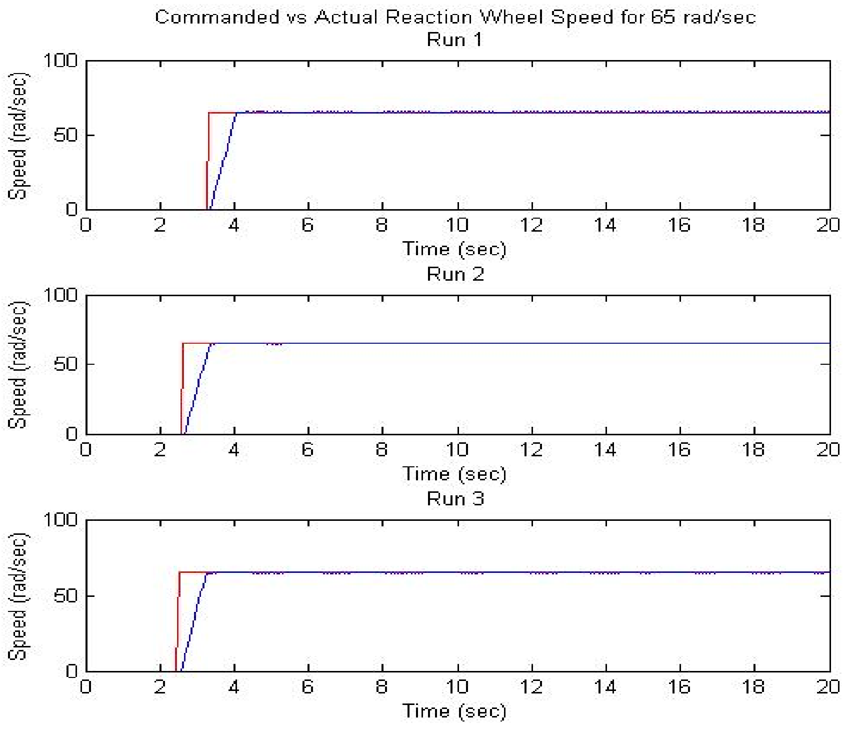

Commanded vs Actual Reaction Wheel Speed for 40 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)
0
2
4
6
8
10
12
14
16
18
20
20
40
60

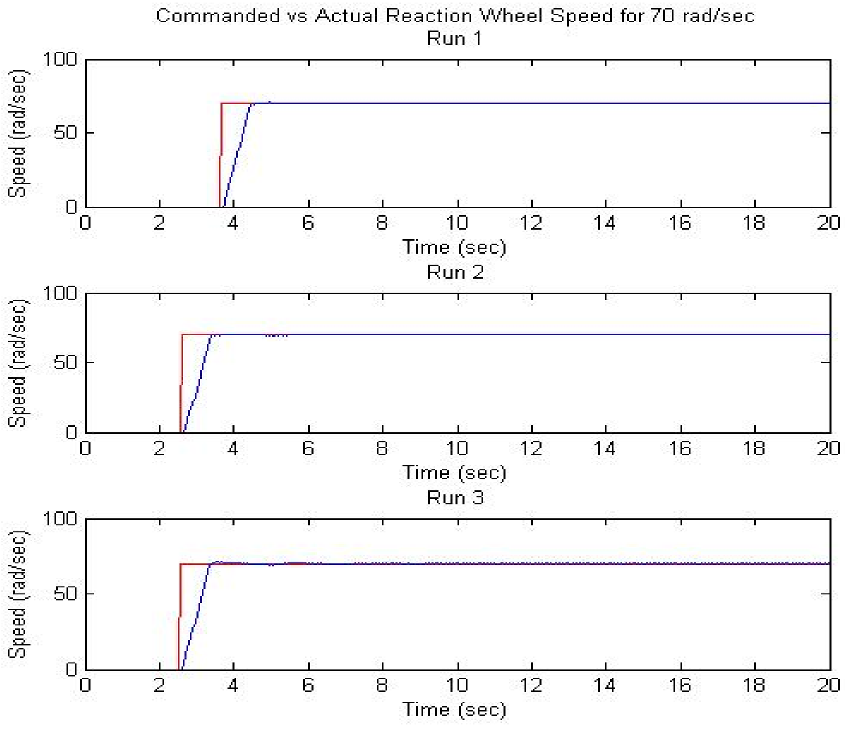

Commanded vs Actual Reaction Wheel Speed for 45 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)
0
2
4
6
8
10
12
14
16
18
20
0
20
40
60

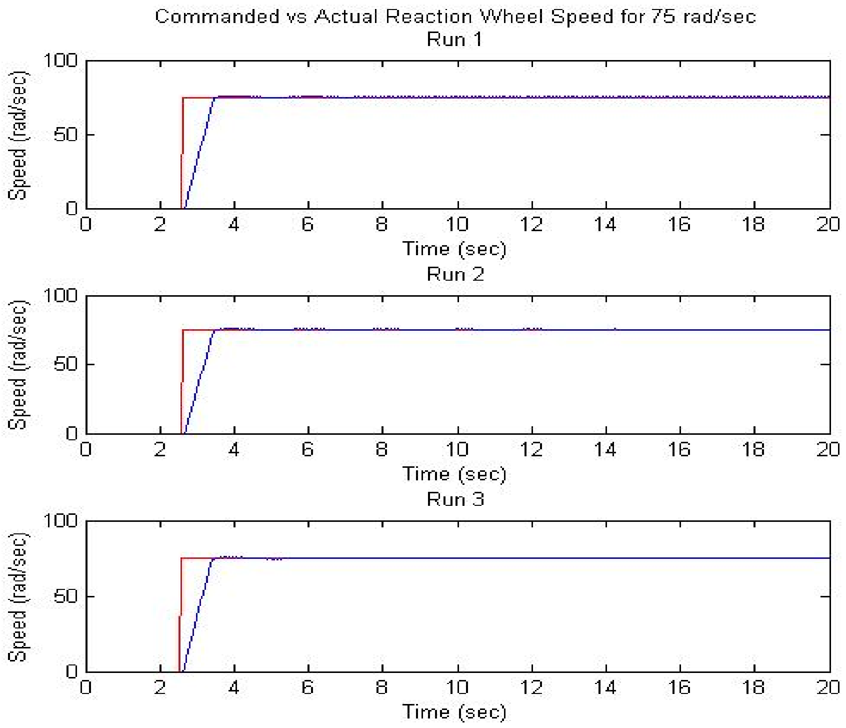

Commanded vs Actual Reaction Wheel Speed for 50 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)
0
2
4
6
8
10
12
14
16
18
20
0
20
40
60

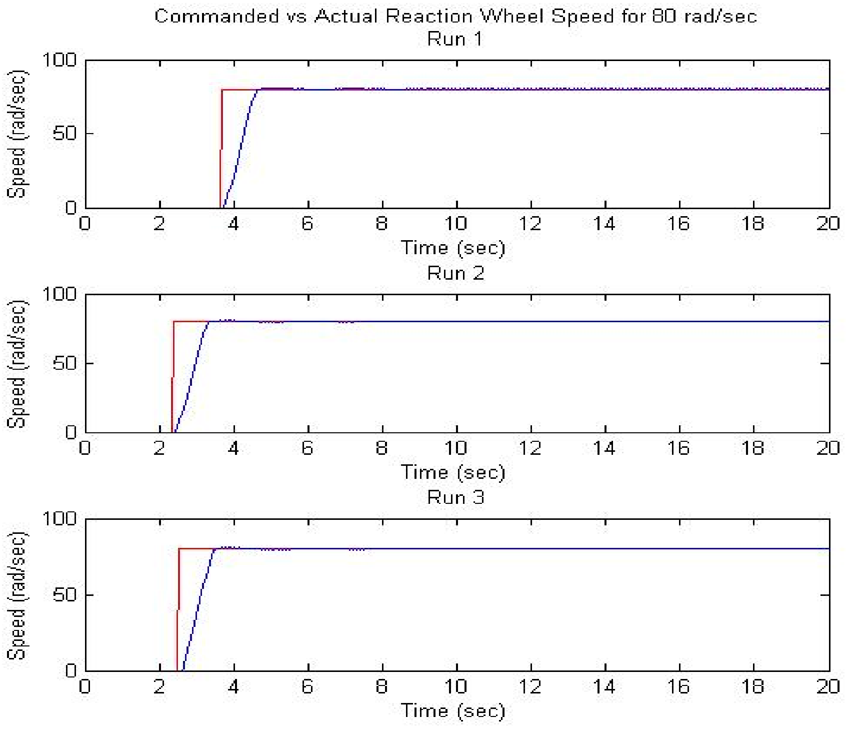

Commanded vs Actual Reaction Wheel Speed for 55 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)
0
20
40
60
2
4
6
8
10
12
14
16
18
20


Commanded vs Actual Reaction Wheel Speed for 60 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)
0
50
100
2
4
6
8
10
12
14
16
18
20


Commanded vs Actual Reaction Wheel Speed for 65 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)
0
50
100
0
2
4
6
8
10
12
14
16
18
20


Commanded vs Actual Reaction Wheel Speed for 70 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)
0
50
100
0
2
4
6
8
10
12
14
16
18
20


Commanded vs Actual Reaction Wheel Speed for 75 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)
0
2
4
6
8
10
12
14
16
18
20
0
50
100

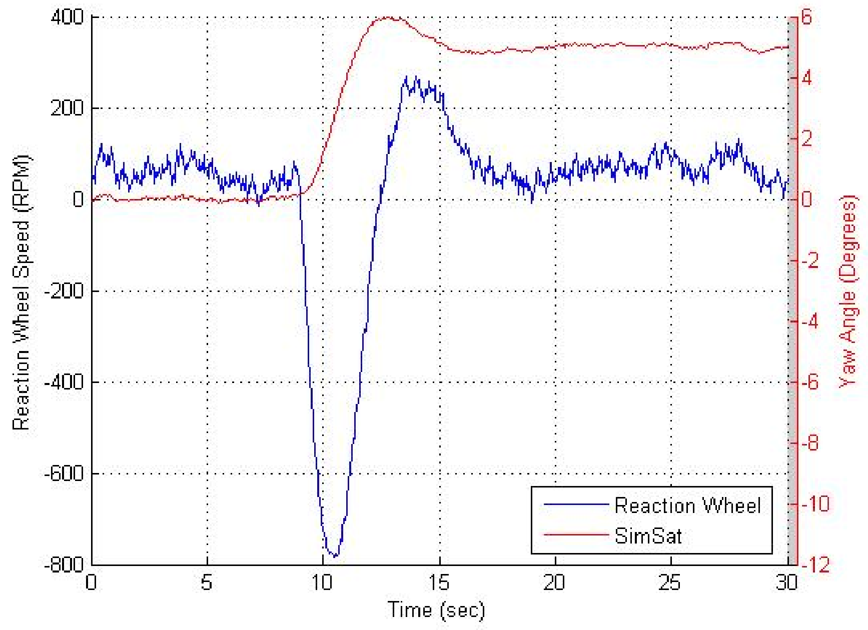

Commanded vs Actual Reaction Wheel Speed for 80 rad/sec
Run 1
Run 2
Run 3
Speed (rad/sec)
Time (sec)
0
2
4
6
8
10
12
14
16
18
20
0
50
100

Appendix F: Matlab Code to Create SimSat Orientation Change Plots

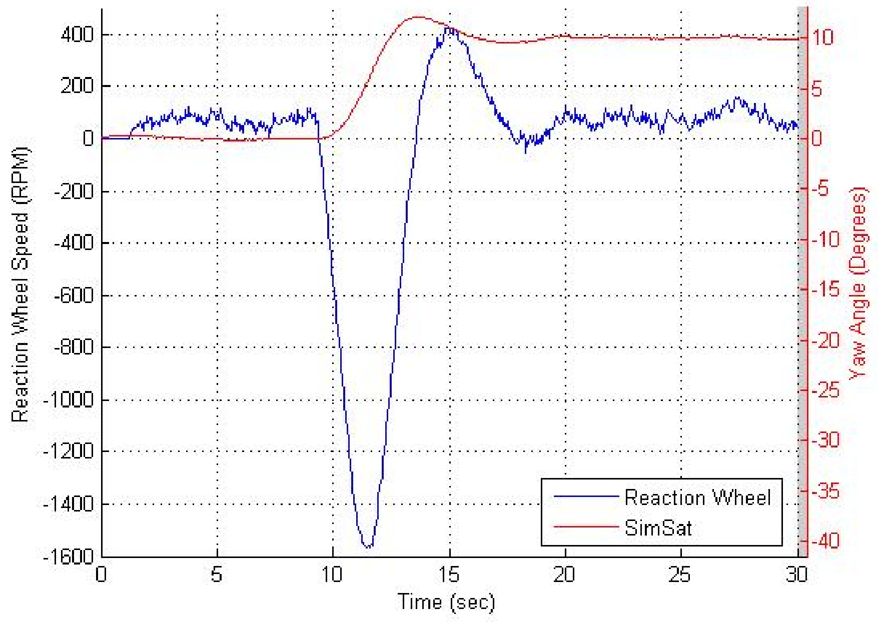
```
% Closed Loop Plots of SimSat
% Created by Capt Jason Geitgey
% June 06
% Open source as long as credit is given
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script takes data collected from the SimSat closed loop model and data
% captured using control desk to generate plots of the reaction wheel speed
% of SimSat and the resulting yaw orientation.  These plots are generated
% from data extracted from the control desk generated files.  This file will
% also compare the open loop model data compared to the actual hardware data
% captured from SimSat.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all; clc;
close all;

% Load the file into the workspace and assign it to a variable called data
% to make it easy to work with no matter the file name

%load yaw_5deg
%load yaw_10deg
%load yaw_25deg
%load yaw_45deg
load yaw_90deg

% Change this name to match the data loaded in the workspace

%data = yaw_5deg
%data = yaw_10deg
%data = yaw_25deg
%data = yaw_45deg
data = yaw_90deg

% Load the data against variable names

t = data.X.Data';                       % time increment for plots (sec) in .05 step increments due
                                        %to model settings
y1_raw = data.Y(1,1).Data';             % From PD control basic commanded wheel rate for SimSat (rad/sec)
y2_raw = data.Y(1,2).Data';             % position error
y3_raw = data.Y(1,3).Data';             % Actual wheel speeds measured from SimSat (rad/sec)
y4_raw = data.Y(1,4).Data';             % SimSat yaw angle (rad)
y5_raw = data.Y(1,5).Data';             % Measured total commanded wheel rate for SimSat following
                                        % addition block in reaction wheel block (rad/sec)
y6_raw = data.Y(1,6).Data';             % SimSat yaw rate (rad/sec)

% Convert data to be in RPM and degrees instead of raw SimSat telemetry data

basic_rw_cmd = y1_raw*(60/(2*pi));   %Convert to RPM from rad/sec
pos_error = y2_raw*(180/pi);         %Convert to degrees from rad
rw_speed = y3_raw*(60/(2*pi));       %Convert to RPM from rad/sec
yaw_ang = y4_raw*(180/pi);           %Convert to degrees from rad
rw_cmd = y5_raw*(60/(2*pi));         %Convert to RPM from rad/sec
```

```
yaw_rate = y6_raw*(60/(2*pi));   %Convert to RPM from rad/sec

% Plot the reaction wheel speed vs the resulting Yaw angle of SimSat from
% the collected hardware data.  Uses a special two axis script found at
% Matlab file exchange

figure(1)
plot(t,rw_speed, 'b');
ylim([-2300,2200]);
addaxis(t,yaw_ang,[-104,100],'r');
grid on
xlabel('Time (sec)');
addaxislabel(1,'Reaction Wheel Speed (RPM)');
addaxislabel(2,'Yaw Angle (Degrees)');
legend('Reaction Wheel','SimSat','location', 'SE');
%legend boxoff
%{
% Plot to compare the actual hardware vs model data
figure(2)
plot(t, yaw_ang, 'b', t, yaw_angle_sim*(180/pi), 'r');
title('Comparison of SimSat Yaw Angle to Model Yaw Angle')
xlabel('Time (sec)')
ylabel('Yaw Angle (Degrees)')
%End of file
%}
```

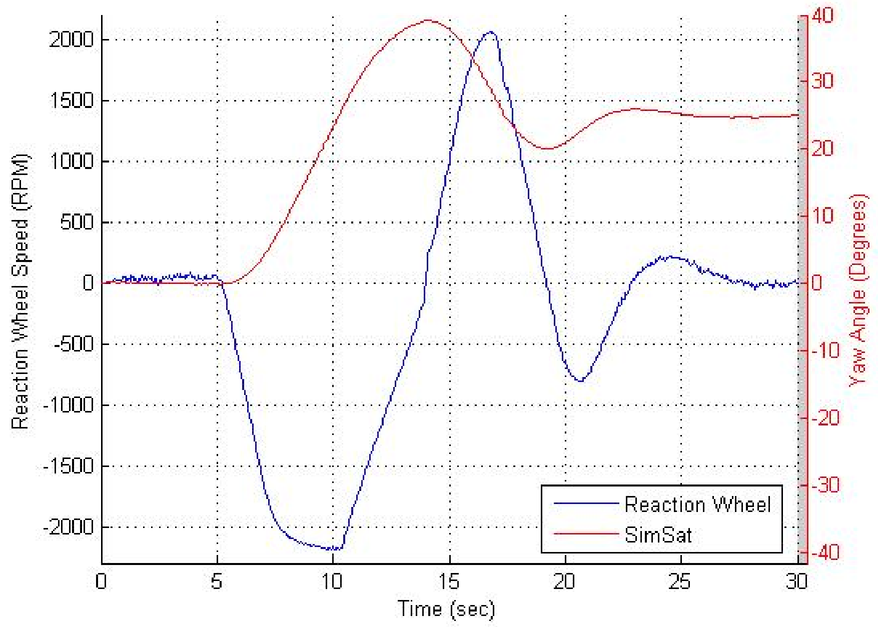
Appendix G: SimSat Hardware Orientation Change Plots



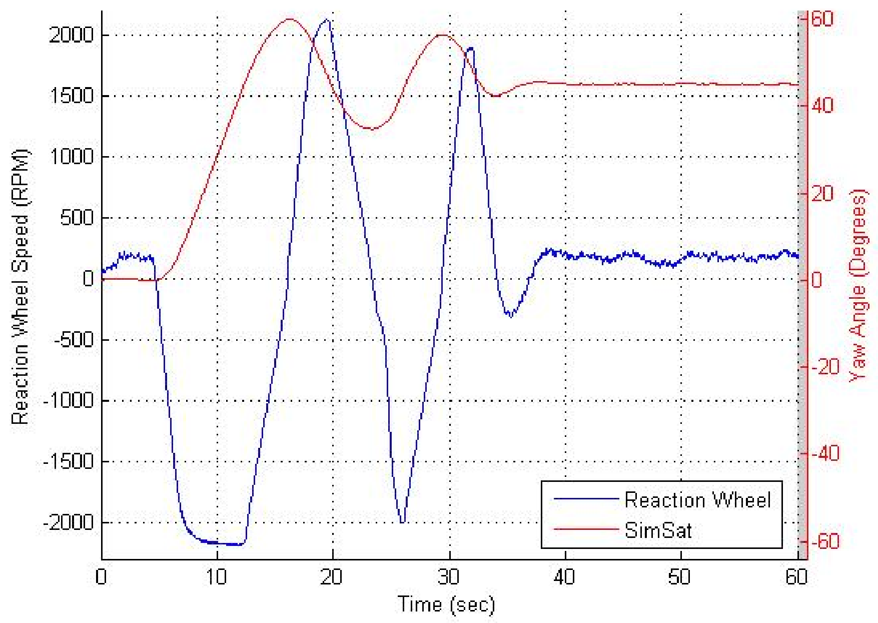
Plot of SimSat Closed Loop Hardware Reaction Wheel Speed versus Yaw Angle for a 5° Commanded Change from Rest



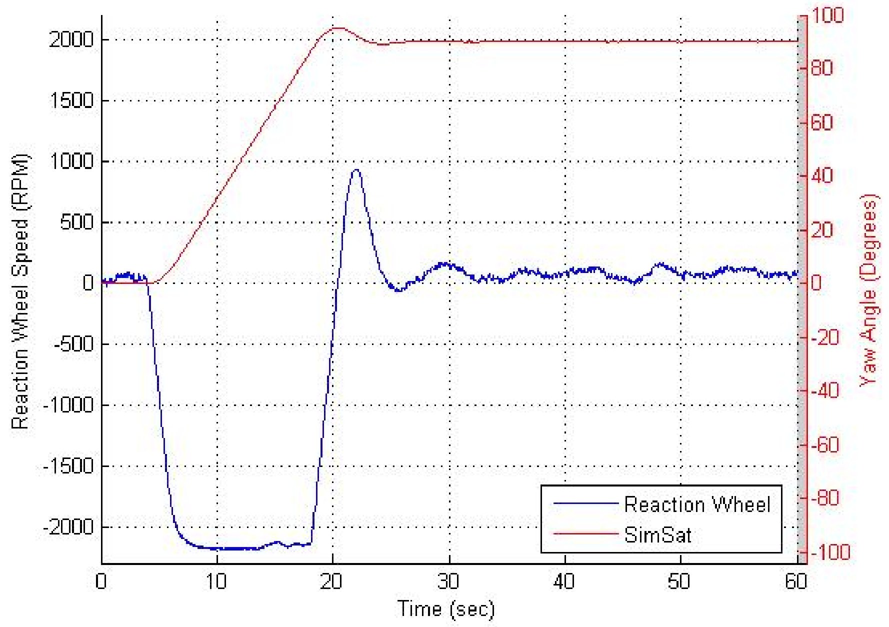
Plot of SimSat Closed Loop Hardware Reaction Wheel Speed versus Yaw Angle for a 10° Commanded Change from Rest



Plot of SimSat Closed Loop Hardware Reaction Wheel Speed versus
Yaw Angle for a 25º Commanded Change from Rest



Plot of SimSat Closed Loop Hardware Reaction Wheel Speed versus
Yaw Angle for a 45º Commanded Change from Rest



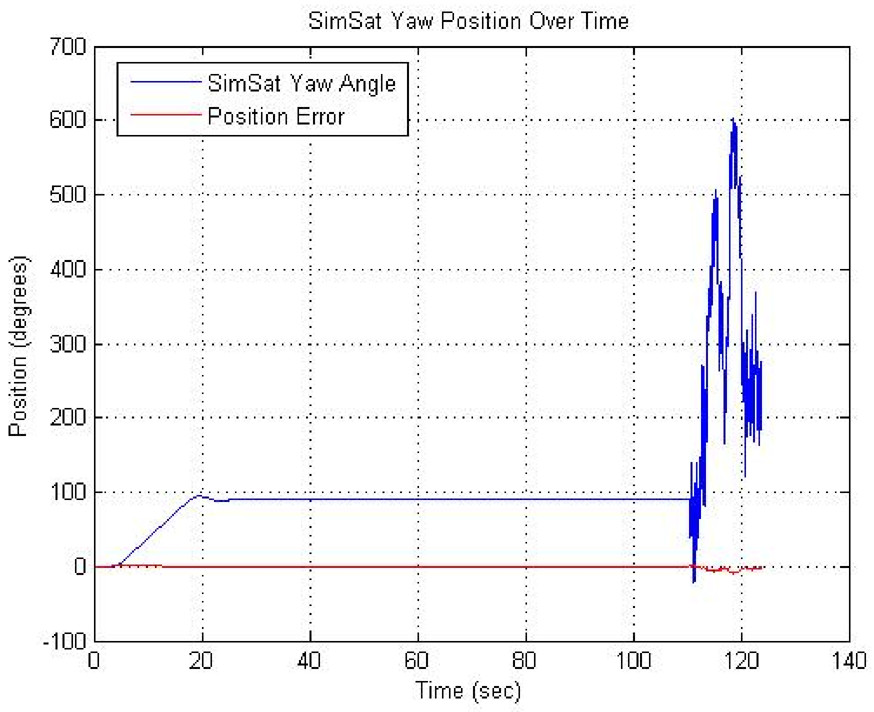
Plot of SimSat Closed Loop Hardware Reaction Wheel Speed versus Yaw Angle for a 90º Commanded Change from Rest

Appendix H: Find Hardware MOI Matlab® Code

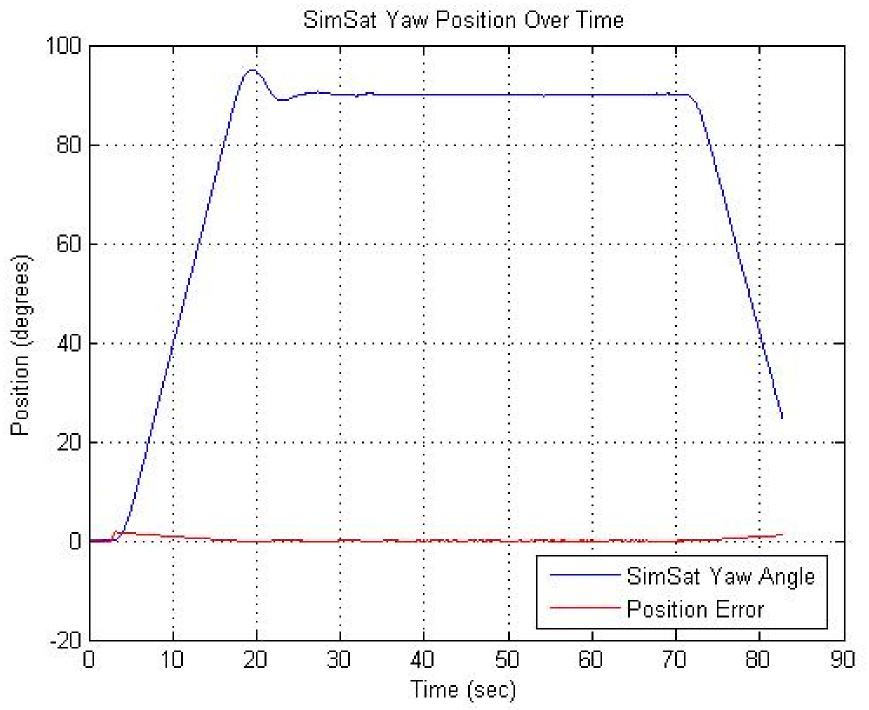
```
% Determine the MOI of SimSat
% Written by Capt Jason Geitgey
% June 2006
% Open source as long as credit is given

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This code is designed obtain the MOI of SimSat using the already tuned
% baseline math model of SimSat.  The baseline is tuned from obtaining
% MOI values using the open loop method and finding the appropriate PD gains
% that match the baseline data.  Then keeping everything fixed this code
% is run on different configurations of SimSat to determine the change
% in MOI from the baseline.  The unknown orientation data is loaded and is
% compared to the data gained from the math model the only thing that is of
% concern is making sure the start times sync between the model and
% hardware data.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% This code is for one direction only and currently configured for the
% yaw direction using SimSat

clear all; clc; close all;
format long e;

display('Code is running to solve problem');

% Declare global variables used in the two scripts

global yaw_data I_sat I_sat_history J_history

% Load the data set that this program will use to find the MOI that
% corresponds to the loaded data.  Data can be from either SimSat hardware
% or the math model.


%load 5_deg_med_1
load 5_deg_med_2
%load 5_deg_med_3
%load 5_deg_base_1
%load 5_deg_base_2
%load 5_deg_base_3


data = deg_med_2
%data = deg_base_1


t = data.X.Data';                 % time increment for plots (sec)

yaw_rw = data.Y(1,7).Data';     % Actual wheel speeds measured from Simsat (rad/sec)
yaw_data = data.Y(1,8).Data';   % SimSat yaw angle (rad)

% Input initial guess for starting the iteration process to determine
% the MOI that matches the loaded data set from either SimSat or
% Simulink math model

I_sat = 10;      %SI Units for MOI are kg*m^2

% Pass the initial guess into a matrix of initial variable in order to pass
% that information into the minimization function

x_init = I_sat;

% Options variable designed to change the tolerance of the function as it
% searches for the MOI and also display the number of iterations of the
% minimization function and the corresponding J cost.  Also I have the
% guess for each iteration of the function output to the screen.

options=optimset('Display','iter','TolFun',1e-6, 'TolX',1e-6);

% Call the minimization function and define x to be the output from the
% completed function and J as the cost function

[x,J] = fminsearch('find_SimSat_MOI_Min_fun',x_init,options);

% Display the obtained value for the MOI that corresponds to the loaded data
% and the minimized J function that corresponds that that MOI

sprintf('Obtained MOI for given yaw data set I_sat = %g',x)
sprintf('Minimized cost function J that corresponds to obtained MOI J = %g',J)

% Plot of each iteration of MOI and cost function.  Should be a bell
% shaped curve or V depending on cost function used and MOI should
% correspond to lowest obtained J.  This is what is printed out to the
% screen above

plot(I_sat_history, J_history, '.');

display('Code completed');
function J = find_SimSat_MOI_Min_fun(x_init)

% Function that is called to determine the MOI for a loaded set of data
% Corresponding global variable declaration

global yaw_data I_sat I_sat_history J_history

% Calling the initial guess passed into the function the variable name that
% is required in the Simulink model being run

I_sat = x_init;

% Running the Simulink/math model in order to obtain the corresponding data
% for the initial guess

[tout,x,y_model] = sim('CLEAN_SIMSAT_MODEL_MINIMIZE',[],[],[]);

% Cost function designed to determine the error between the loaded data and
% the obtained data from the above Simulink/math model


J = sum(abs(yaw_data-y_model).^2);  %My function

% Matrices designed to capture each iteration of the function to be used
% for plotting and shows a history of the functions iterations and
% corresponding J function not needed to work, but nice to have

I_sat_history = [I_sat_history;I_sat];
J_history = [J_history;J];

% This code will iterate each time on the I_sat until the J function is
% minimized.  The I_sat in this file is required for the function to keep
% changing the guess, and the model needs the global variable defined in
% order to obtain the changing I_sat.  The simulink model has to have the
% global variable in order to see it and run properly, if the global
% variable I_sat is not there it will just run using the initial guess and
% not capture the function changing the iterations.
% Written by Capt Jason Geitgey
% June 2006
% Open source as long as credit is given
```

Appendix I: Drift Rate Plots

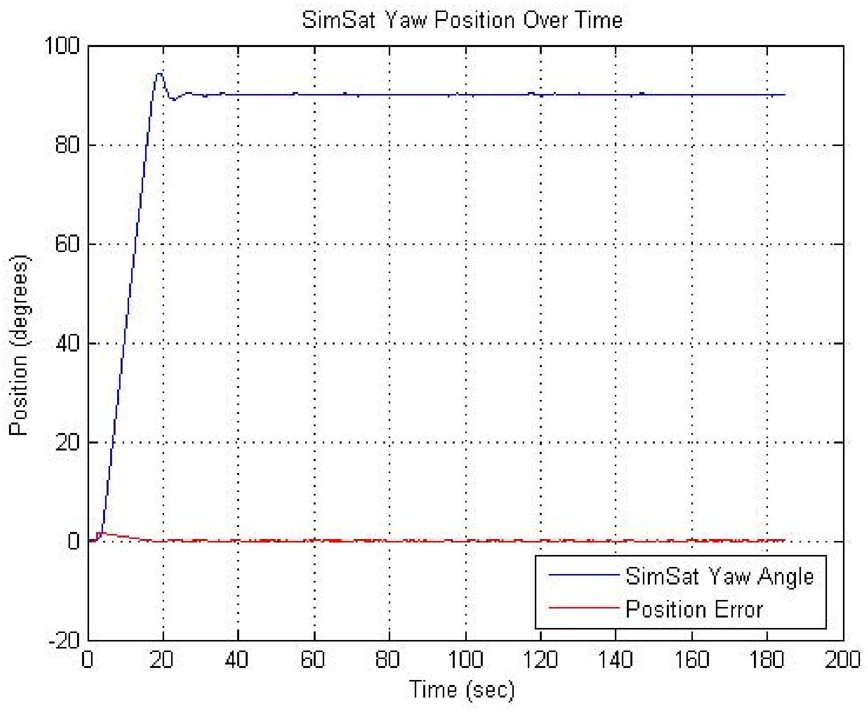
```
% Plots for Drift Rate
% Written by Capt Jason Geitgey
% June 06
% Open source as long as credit is given

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This code is designed to plot SimSat's orientation for the given amount of
% time in the file.  This data was created by commanding SimSat from an at
% rest position to +90 degrees and to hold that position.  This is designed
% to provide the drift rate associated with the gyro.  By using the collected
% laser pointer info and the time it took after SimSat reached +90 until the
% system died gives the time.  The resulting change in angle and the time yields
% the drift rate.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all; clc; close all;
format long e;

load gyro_drift_1   %Load the gyro data to get the plots the numbers must match
%load gyro_drift_2  %between the file up here and the variable callouts below
%load gyro_drift_3
%load gyro_drift_4
%load gyro_drift_5
%load gyro_drift_6
%load gyro_drift_7


t = gyro_drift_1.X.Data';                % time increment for data collected
pos_error = gyro_drift_1.Y(1,2).Data';  % position error between desired
                                         % and actual SimSat position
yaw_ang = gyro_drift_1.Y(1,4).Data';    % SimSat position in (rad)

% Convert radians to degrees for yaw angle
yaw_deg = yaw_ang*(180/pi)

figure(1)
plot(t, yaw_deg, 'b', t, pos_error, 'r')
grid on
title('SimSat Yaw Position Over Time')
xlabel('Time (sec)')
ylabel('Position (degrees)')
legend('SimSat Yaw Angle','Position Error', 'location','SE')% Plots for Drift Rate
```



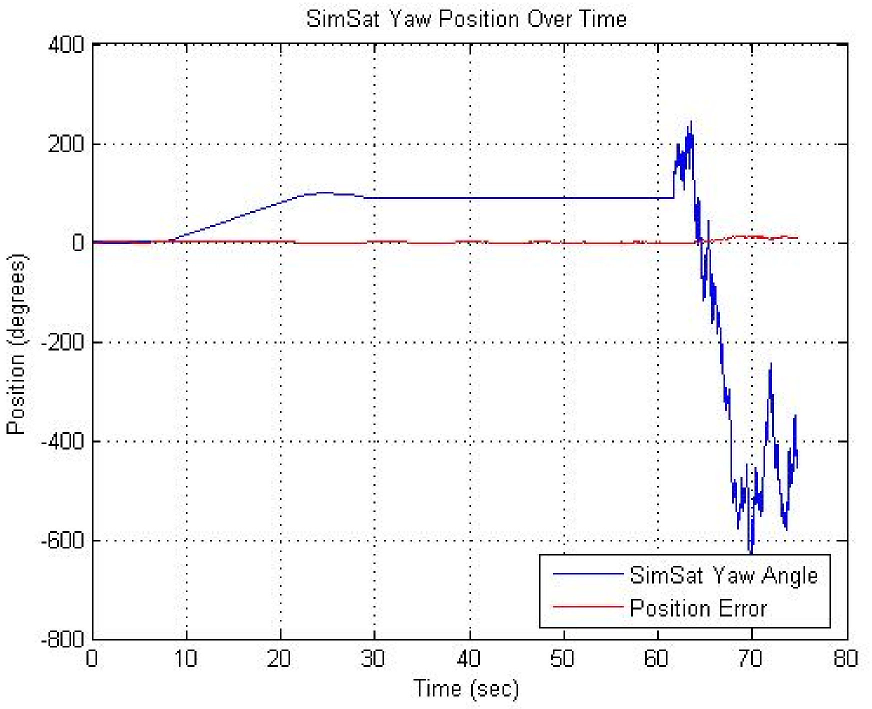
Plot of Yaw Position versus Time for Data Run #1



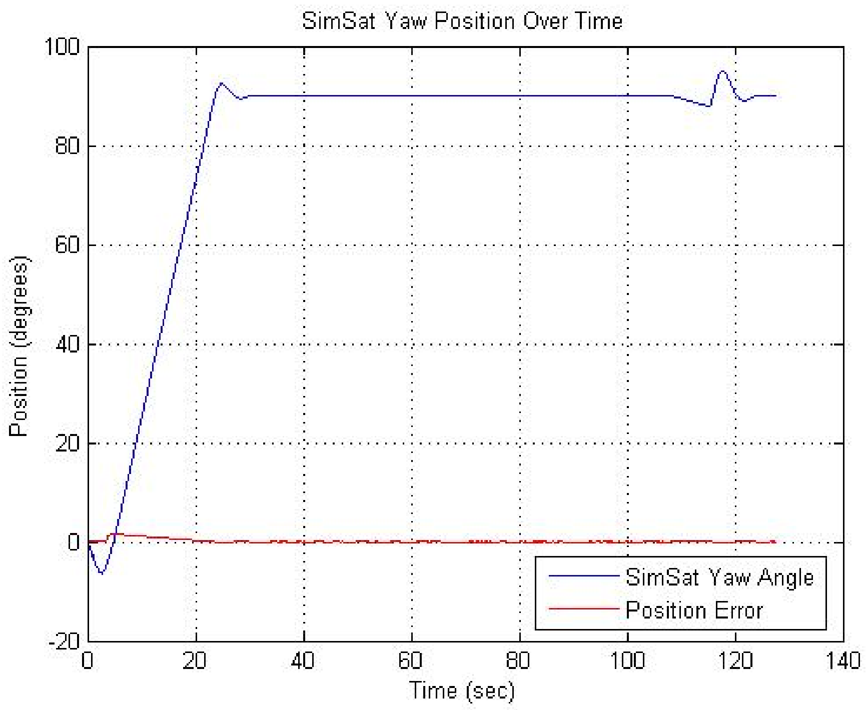
Plot of Yaw Position versus Time for Data Run #2



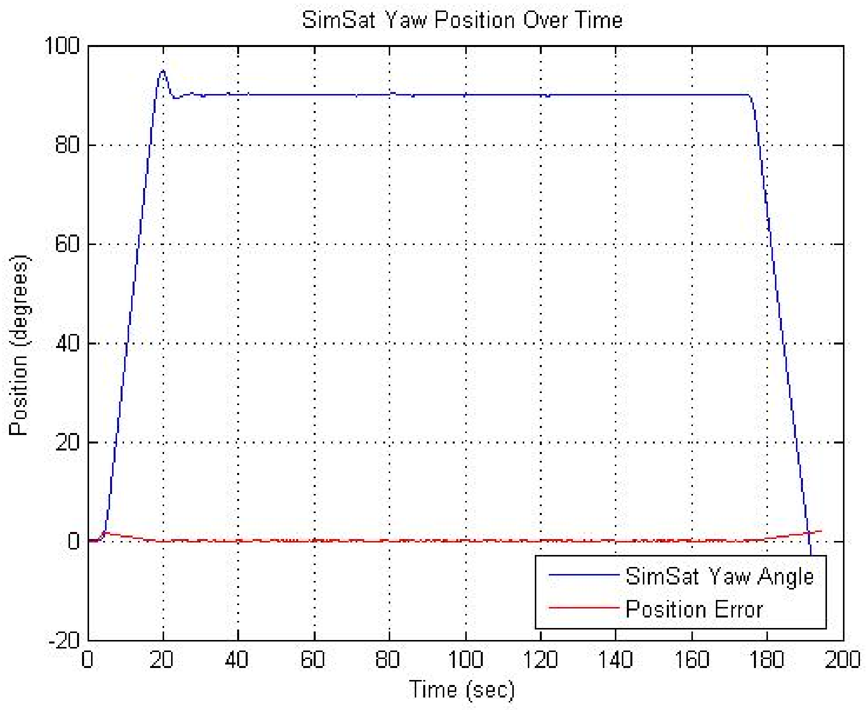
Plot of Yaw Position versus Time for Data Run #3



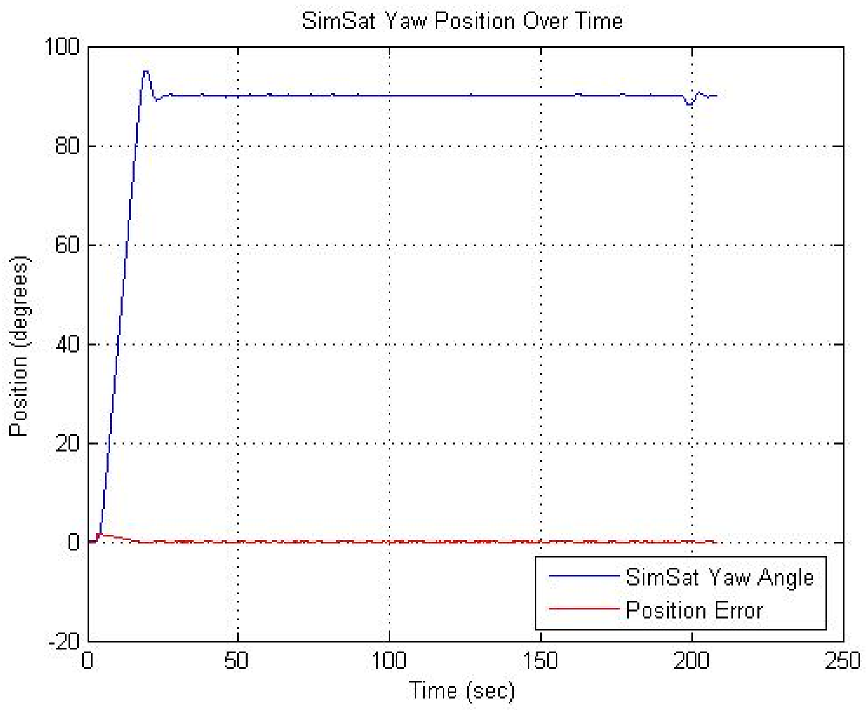
Plot of Yaw Position versus Time for Data Run #4



Plot of Yaw Position versus Time for Data Run #5



Plot of Yaw Position versus Time for Data Run #6



Plot of Yaw Position versus Time for Data Run #7

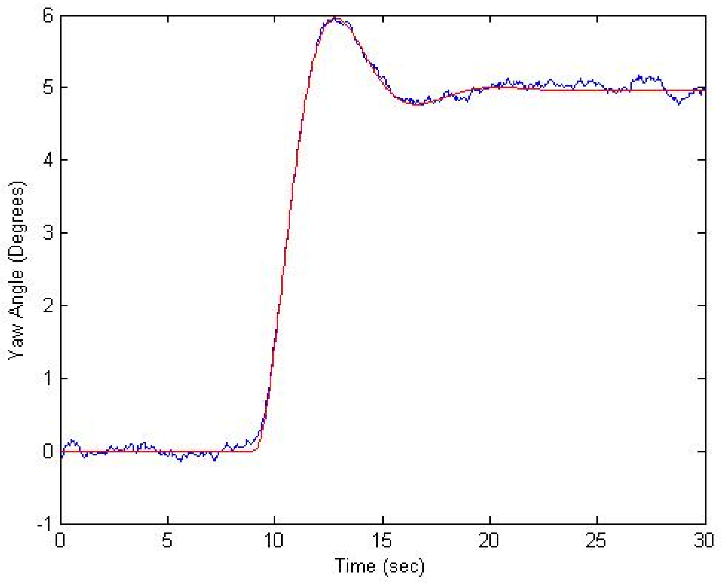
Appendix J: Find PD Gains Matlab® Code

```
% Minimization routine to find PD gain values
% Written by Capt Jason Geitgey
% June 2006
% Open source as long as credit is given
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This code is designed to use the simsat_w_pd_model_minimization simulink
% file to calculate the optimized values Kp and Kd for the PD controller.
% This file uses the SimSat hardware data to compare against the model obtained
% yaw orientation data and minimize the cost function.  The time settings for
% the step command and the total time for the model must match the given data
% or the minimization routine will not work.  This file passes the initial
% guesses of Kp and Kd to a cost function in order to minimize and iterate
% on those values.  It then returns the optimized values.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% This code is designed to only obtain the values for one orientation axis on
% SimSat and is currently configured for the yaw direction

clear all; clc; close all;
format long e;

display('Code is running to solve problem');

% Declare global variables used in the two scripts

global yaw_data Kp Kd Kp_history Kd_history J_history

% Load the data set that this script will use as the truth data source to
% minimize the cost function against.  Data can be from either SimSat hardware
% or the math model.

%load yaw_5deg                  %Select the file you wish to use
%load yaw_10deg
%load yaw_25deg
%load yaw_45deg
%load yaw_90deg



%data = yaw_5deg                %Select the corresponding data call
%data = yaw_10deg
%data = yaw_25deg
%data = yaw_45deg
%data = yaw_90deg

% Load the data against variable names

time = data.X.Data';            % time increment model
y1_raw = data.Y(1,1).Data';     % From PD control basic commanded wheel rate for Simsat (rad/sec)
y2_raw = data.Y(1,2).Data';     % position error
y3_raw = data.Y(1,3).Data';             % Actual wheel speeds measured from Simsat (rad/sec)
yaw_data = data.Y(1,4).Data';           % Simsat yaw angle (rad)
```

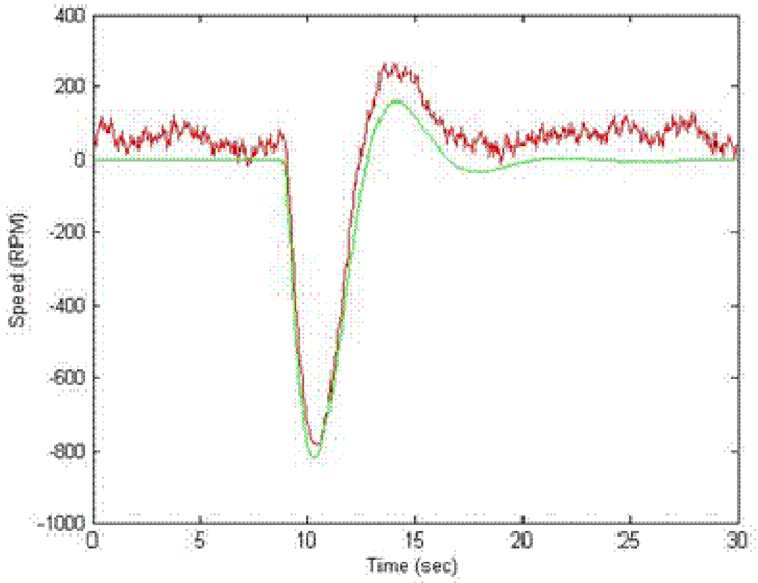
```
y5_raw = data.Y(1,5).Data';              % Measured total commanded wheel rate for Simsat following
                                         addition block in reaction wheel block (rad/sec)
y6_raw = data.Y(1,6).Data';              % Simsat yaw rate (rad/sec)

% Input initial guess for starting the iteration process to determine
% the controller gains that will match the given orientation data

Kp = 1;
Kd = .5;

% Pass the initial guess into a matrix of initial variable in order to pass
% that information into the minimization function

x_init = [Kp Kd];

% Options variable designed to change the tolerance of the function as it
% searches for the controller settings and also display the number of iterations of the
% minimization function and the corresponding J value.

options=optimset('Display','iter','TolFun',1e-6);

% Call the minimization function and define x to be the output from the
% completed function and J as the cost function

[x,J] = fminsearch('find_PD_gains_fun',x_init,options);

% Display the obtained values for Kp and Kd along with the
% minimized J that corresponds to these values

sprintf('Obtained Kp gain value for the given yaw data set Kp = %g',x(1))
sprintf('Obtained Kd gain value for the given yaw data set Kd = %g',x(2))
sprintf('Minimized cost function J that corresponds to obtained gain values J = %g',J)

% Plot of each iteration vs cost function.  Should be a bell
% shaped curve or V depending on cost function used and MOI should
% correspond to lowest obtained J.  This is what is printed out to the
% screen above
%figure(1)
%plot(Kp_history, J_history, '.');
%figure(2)
%plot(Kd_history, J_history, '.');

display('Code completed');

function J = find_PD_gains_fun(x_init)

% Function that is called to determine the PD gains for a loaded set of data
% Corresponding global variable declaration

global yaw_data Kp Kd Kp_history Kd_history J_history

% Calling the initial guess passed into the function the variable name that
% is required in the Simulink model being run

```

```
Kp = x_init(1);
Kd = x_init(2);

% Running the Simulink/math model in order to obtain the corresponding data
% for the initial guess

[tout,x,y_model] = sim('simsat_w_pd_model_minimization',[],[],[]);

% Cost function designed to determine the error between the loaded data and
% the obtained data from the above Simulink/math model


J = sum(abs(yaw_data-y_model).^2);  %My function

% Matrices designed to capture each iteration of the function to be used
% for plotting and shows a history of the functions iterations and
% corresponding J function not needed to work, but nice to have

Kp_history = [Kp_history; Kp];
Kd_history = [Kd_history; Kd];
J_history = [J_history; J];

% This code will iterate each time on Kp and Kd until the J function is
% minimized.  The Kp and Kd in this file is required for the function to keep
% changing the guess, and the model needs the global variable defined in
% order to obtain the changing Kp and Kd.  The simulink model has to have the
% global variable in order to see it and run properly.
% Written by Capt Jason Geitgey
% June 2006
% Open source as long as credit is given
```

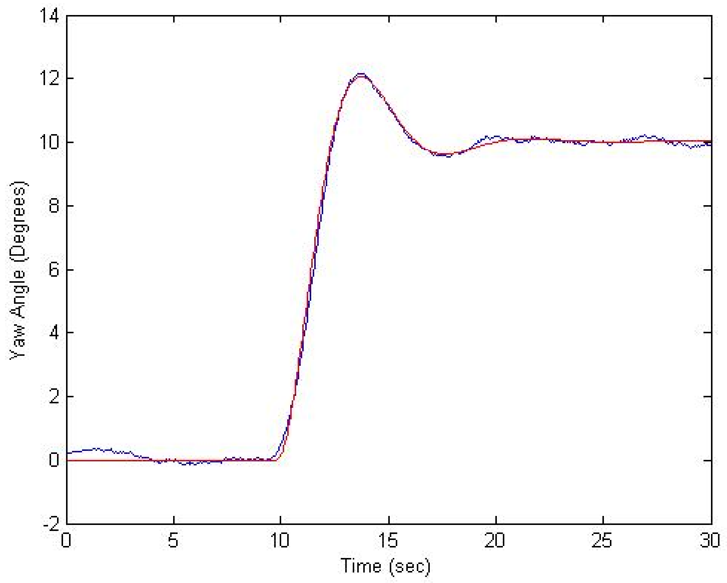
Appendix K: Optimized PD/RW SimSat Model Plots



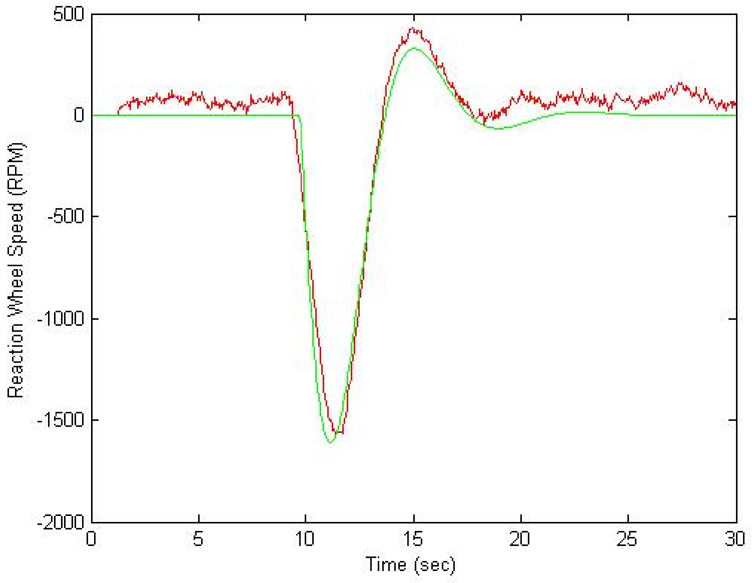
Comparison of Hardware vs Optimized PD/RW Model for 5º Orientation Change



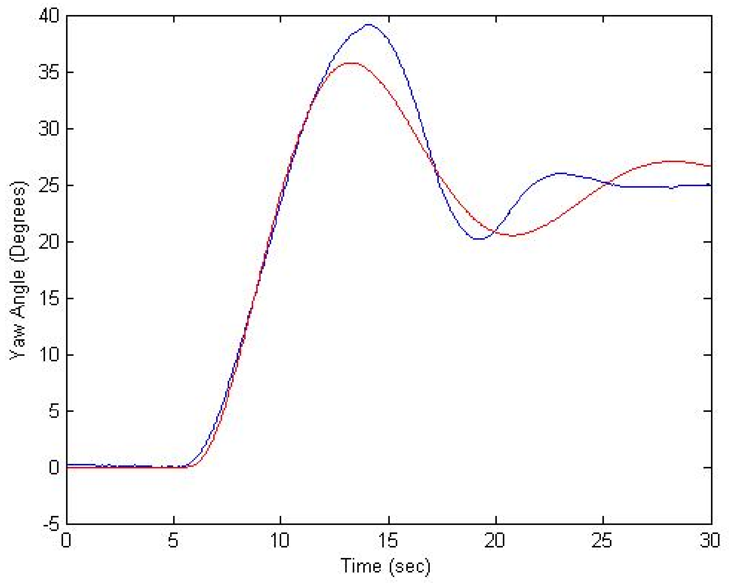
Comparison of Hardware vs Model Reaction Wheel Speed for 5º Orientation Change



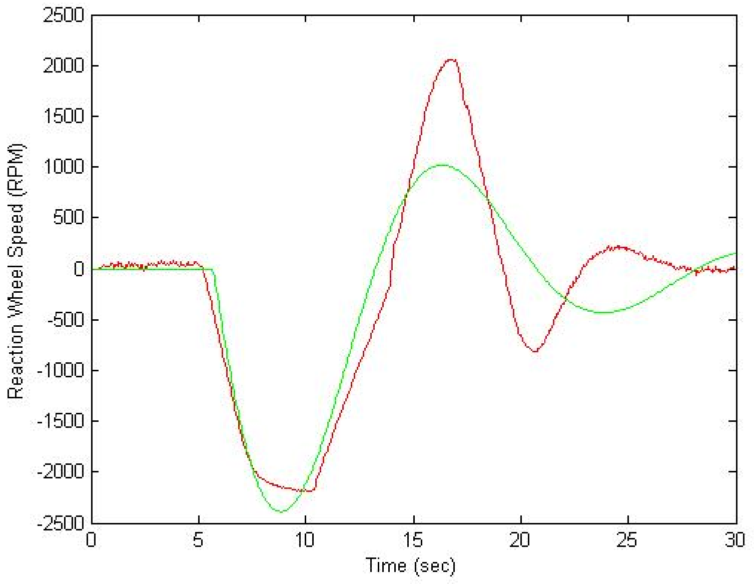
Comparison of Hardware vs Optimized PD/RW Model for 10º Orientation Change



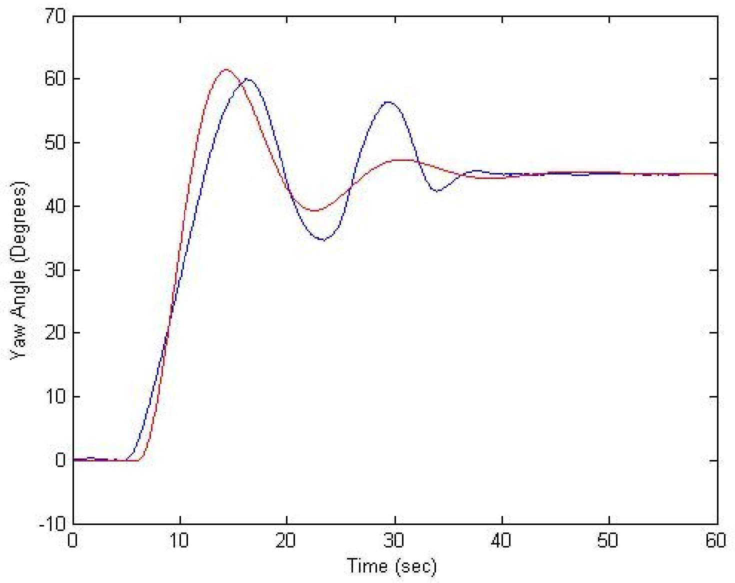
Comparison of Hardware vs Model Reaction Wheel Speed for 10º Orientation Change



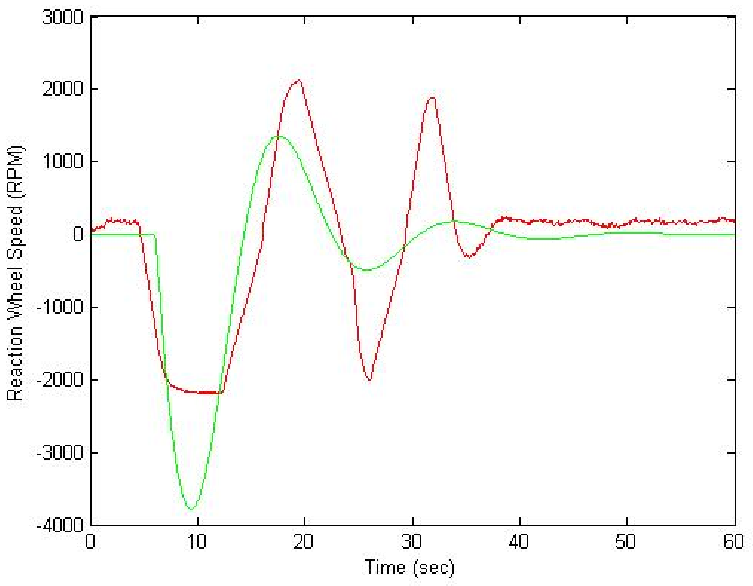
Comparison of Hardware vs Optimized PD/RW Model for 25º Orientation Change



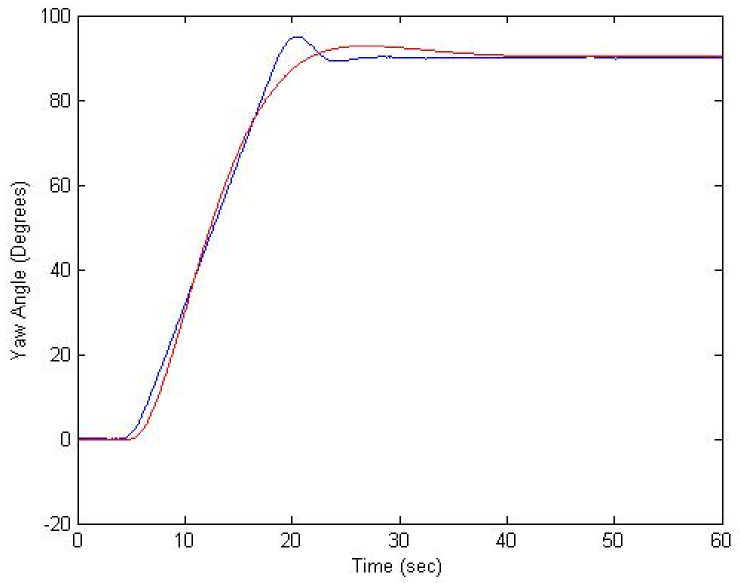
Comparison of Hardware vs Model Reaction Wheel Speed for 25º Orientation Change



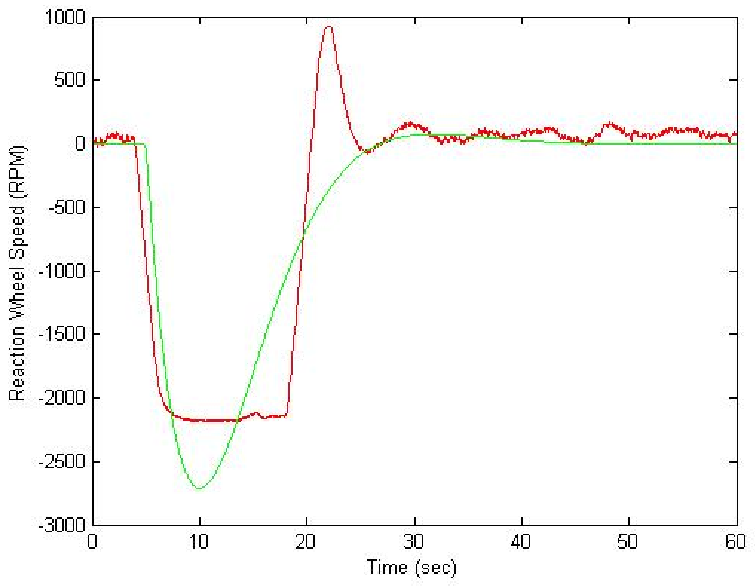
Comparison of Hardware vs Optimized PD/RW Model for 45º Orientation Change



Comparison of Hardware vs Model Reaction Wheel Speed for 45º Orientation Change



Comparison of Hardware vs Optimized PD/RW Model for 90º Orientation Change



Comparison of Hardware vs Model Reaction Wheel Speed for 90º Orientation Change

Bibliography

Baruh, Haim. *Analytical Dynamics*. McGraw-Hill, Inc., 1999.

Bergmann, E. V., B. K. Walker, and D. R.Levy. "Mass Property Estimation for Control of Asymmetrical Satellites," *Journal of Guidance, Control, and Dynamics,* 10: 483-491 (September-October 1987).

Bergmann, E. and J. Dzielski. "Spacecraft Mass Property Identification with Torque-Generating Control," *Journal of Guidance, Control, and Dynamics,* 13: 99-103 (January-February 1990).

Bryson, Arther E. Jr. *Dynamic Optimization*. Menlo Park CA. Addison Wesley Longman, Inc., 1999.

Colebank, James E., Robert D. Jones, George R. Nagy, Randall D. Pollak, and Donald R. Mannebach. *SIMSAT: A Satellite System Simulator and Experimental Test Bed for Air Force Research.* MS thesis, AFIT/GSE/GSO/ENY/99M-1. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, March 1999.

Cooper, J. E. and J. R. Wright. "Spacecraft In-Orbit Identification Using Eigensystem Realization Methods," *Journal of Guidance, Control, and Dynamics,* 15: 352-359 (March-April 1992).

Dabrowski, Vincent J. *Experimental Demonstration of an Algorithm to Detect the Presence of a Parasitic Satellite.* MS thesis, AFIT/GAE/ENY/03-02. School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, March 2003.

De Selding, Peter B. "Amazonas Propellant Tank Suffers Sharp Drop in Pressure." Press Release. http://www.space.com/spacenews/archive04/amazonasarch_091004.html, Imaginova Corp, 7 January 2006.

Franklin, Gene F., J. David Powell, and Abbas Emami-Naeini. *Feedback Control of Dynamic Systems* (3rd Edition). Addison-Wesley Publishing Company, Inc., 1994.

French, David B. *Hybrid Control Strategies for Rapid, Large Angle Satellite Slew Maneuvers.* MS thesis, AFIT/GA/ENY/03-02. School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, March 2003.

Goldin, Daniel S., Administrator National Aeronautics and Space Administration. "Affordable Access to Space." Statement before the Subcommittee on Science, Technology, and Space Committee on Commerce, Science, and Transportation. United States Senate, Washington D.C.. 23 September 1998

Halfman, Robert L. *Dynamics.* Addison-Wesley Publishing Company, Inc., 1962.

Kimsal, Matthew B. *Design of a Space-Born Autonomous Infrared Tracking System.* MS thesis, AFIT/GA/ENY/04-M02. School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, March 2004.

Matlab. Version 7.0.4 Release 14, Service Pack 2. Computer software. MathWorks, Inc, Natick MA, 2005

Ogata, Katsuhiko. *Modern Control Engineering* (4th Edition). Prentice Hall, Inc., 2002

Peck, Mason A. "Mass-Properties Estimation for Spacecraft with Powerful Damping," *Proceedings of the AAS/AIAA Astrodynamics Specialists Conference.* Quebec City, Canada, August 2001.

-----. "Estimation of Inertia Parameters for Gyrostats Subject to Gravity Gradient Torques," *Proceedings of the AAS/AIAA Astrodynamics Specialists Conference.* Girdwood, AK, August 1999.

Smith, Jason E. *Attitude Model of a Reaction Wheel/ Fixed Thruster Based Satellite Using Telemetry Data.* MS thesis, AFIT/GA/ENY/05-M010. School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, March 2005.

Tanygin, Sergei and Trevor Williams. "Mass Property Estimation Using Coasting Maneuvers," *Journal of Guidance, Control, and Dynamics*, 20: 625-632 (July-August 1997).

Venere, Emily. "Technique Detects When Satellites are Low on Fuel." Press release. http://www.purdue.edu/UNS/html4ever/0104.Collicott.satellites.html, Purdue University, 7 January 2006.

Walter, Dan. "RE: Info on Satellite Costs." Electronic Mail from daniel.walter@losangeles.af.mil to Geitgey, Jason, AFIT/ENY. April 2006.

Wertz, Julie A. and Allan Y. Lee. "In-Flight Estimation of the Cassini Spacecraft's Inertia Tensor," *Journal of Spacecraft and Rockets,* 39: 153-155 (July-August 1997).

Wiesel, William E. *Spaceflight Dynamics* (2nd Edition). McGraw-Hill, Inc., 1997.

Vita

Captain Jason W. Geitgey was raised in Northwest Ohio and graduated from Defiance High School. He entered undergraduate studies at the University of Cincinnati in Cincinnati, Ohio where he graduated with a Bachelor of Science degree in Aerospace Engineering in September 1998 along with being commissioned through Detachment 665 AFROTC.

His first assignment was at Wright-Patterson AFB, Ohio as an Acquisition Program Manager Intern at the Aeronautical System Center. As part of his training, he spent one year as a Propulsion Research and Development Acquisition Logistics Engineer in the Air Force Research Lab's Propulsion Directorate. He was then assigned to the Propulsion Development System Office as the F-22 and Component Improvement Program Test manager. There, he was responsible for over $75M in engine test and development activities for various fighter and transport aircraft. In November 2001, he was assigned to Air Mobility Command's Test and Evaluation Squadron at Fort Dix, New Jersey, where he served as a flight test engineer, executive officer, and assistant flight commander. He was responsible for development and execution of command specific force development and evaluation programs on transport and tanker aircraft. In August 2004, he entered the Graduate School of Engineering and Management, Air Force Institute of Technology. Upon graduation, he will be assigned to the United States Air Force Test Pilot School, Edwards AFB, California.

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)* 13 Jun 2006	**2. REPORT TYPE** **Master's Thesis**	**3. DATES COVERED** *(From – To)* Aug 2004 – June 2006

4. TITLE AND SUBTITLE The Determination of Remaining Satellite Propellant Using Measured Moments of Inertia	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR Geitgey, Jason, W., Captain, USAF	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAMES AND ADDRESS Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT/GAE/ENY/06-J04
9. SPONSORING/MONITORING AGENCY NAME AND ADDRESS N/A	**10. SPONSOR/MONITOR'S ACRONYM**
	11. SPONSOR/MONITOR'S REPORT NUMBER

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This research is designed to demonstrate that a change in satellite propellant can be determined using measured moments of inertia (MOI) from a satellite. Because satellites are currently incapable of being refueled in orbit it is important to have multiple methods to determine the remaining fuel onboard. This research can also support satellite operator selection of control-system gains to improve performance or recover the spacecraft. To meet the research objectives, new mathematical models of the Air Force Institute of Technology's Simulated Satellite (SimSat) were developed. These models were created using dynamic response analysis techniques on the reaction wheel and SimSat systems. The models were than validated against the existing SimSat hardware. Using a least-squares parameter estimation technique, the model and hardware data were compared to determine the resulting change in measured MOI. Then, using a calibrated baseline model, telemetry data was compared to the model to determine the MOI of the unknown system. The research found it is possible to determine the change in satellite fuel from measured MOI. The research also found there are limits to this detection technique based on the accuracy of the mathematical model and the angle of the detection maneuver being performed.

15. SUBJECT TERMS
Satellite, Orbiting Satellite, Moment of Inertia, MOI, Simulated Satellite, SimSat, Mathematical Models, Propellant, Optimization, Least Squares Method

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Dr. Richard G. Cobb - ENY
REPORT U	**ABSTRACT** U	**c. THIS PAGE** U	UU	163	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-3636, ext 4559 e-mail: richard.cobb@afit.edu